UNIVERSIDADE DO EXTREMO SUL CATARINENSE - UNESC

CURSO DE CIÊNCIA DA COMPUTAÇÃO

RICARDO LINEBURGER MONDARDO

O ALGORITMO C4.5 NA TAREFA DE CLASSIFICAÇÃO NA *SHELL* ORION DE *DATA MINING ENGINE*

CRICIÚMA, NOVEMBRO DE 2009

RICARDO LINEBURGER MONDARDO

O ALGORITMO C4.5 NA TAREFA DE CLASSIFICAÇÃO

NA *SHELL* ORION DE *DATA MINING ENGINE*

Trabalho de Conclusão de Curso apresentado para obtenção do Grau de Bacharel em Ciência da Computação da Universidade do Extremo Sul Catarinense.

Orientadora: Profª. MSc. Merisandra Côrtes de Mattos

CRICIÚMA, NOVEMBRO DE 2009

RICARDO LINEBURGER MONDARDO

O ALGORITMO C4.5 NA TAREFA DE CLASSIFICAÇÃO NA *SHELL* ORION DE *DATA MINING ENGINE*

Submetido ao corpo docente do Curso de Ciência da Computação da Universidade do Extremo Sul Catarinense como um dos requisitos para obtenção do grau de Bacharel em Ciência da Computação.

Profa. MSc. Ana Claudia Garcia Barbosa
Coordenadora do Curso de Ciência da Computação

Banca Examinadora:

Profa. MSc. Merisandra Côrtes de Mattos (UNESC)
Orientadora

Prof. MSc. Cristian Cechinel (Universidade Federal de Bagé/RS)

Esp. Anne Marie Scoss (UNESC)

À minha querida família e amigos pelo grande apoio concebido para a conclusão desta pesquisa.

AGRADECIMENTOS

Agradeço em primeiro lugar a Deus por dar força e vontade de vencer nesta etapa dos meus estudos.

Aos meus pais, Eliana e Vanderlei, pelo apoio e carinho concedidos em toda a minha vida e principalmente durante a realização desta pesquisa.

A minha irmã, Aline, e minha namorada, Luciane, por sempre incentivarem e prestarem o suporte necessário na conclusão desta pesquisa.

A minha orientadora, Merisandra, pela compreensão e dedicação concebidas ao longo da pesquisa.

E finalmente, a todos que agregaram conhecimento, de qualquer maneira, para a conclusão desta pesquisa.

“Amor pelos outros e respeito por seus direitos e sua dignidade, sejam eles quem forem ou o que forem: é só o que afinal precisamos ter.”

Dalai-Lama

RESUMO

A tendência da utilização de bases de dados de armazenamento está presente em um grande número das empresas operantes no mercado de trabalho. O objetivo da armazenagem destas informações por tempo prolongado é reter o histórico dos registros contidos nas bases a fim de proporcionar tomadas de decisões em virtude da geração de conhecimento. A necessidade da análise destas informações se torna inviável ao homem apenas com a utilização da visão e do raciocínio lógico, tendo neste ponto o surgimento do *data mining*, que é um processo automatizado de busca de conhecimento em bases de dados. Nesta pesquisa o *data mining* foi empregado a fim de agregar na *Shell* Orion *Data Mining Engine* a tarefa de classificação por meio do algoritmo C4.5 utilizando amostragem dos dados por meio da indução de árvores de decisão. Este algoritmo além de realizar uma análise nos dados gerando um modelo classificador, utiliza cálculos estatísticos na avaliação de seus resultados e verifica se está gerando informações corretas ou não. Assim, os locais geradores de incertezas são excluídos do modelo resultando numa precisão mais adequada ao que se espera. A etapa de implementação e testes da pesquisa realizados, confirmam o conhecimento obtido durante o estudo sobre o assunto, no qual identifica a classificação por meio da indução de árvores de decisão como um processo facilmente compreendido pelo homem, obtendo resultados satisfatórios quanto ao tempo e ao nível de precisão em relação a outras ferramentas de *data mining*.

Palavras-Chave: *Data Mining*; Classificação; Árvores de Decisão; Algoritmo C4.5; *Shell* Orion.

ABSTRACT

The use of databases is present in several companies working in the market. The purpose of storing this information for a long time is to retain the historical records contained in the databases to provide decision-making in order for the generation of knowledge. The need of the information analysis becomes impossible to man, only by the use of vision and logical reasoning, and at this point have the emergence of data mining, which is an automated process of searching for knowledge in databases. In this study the data mining was used to add the Shell Orion Data Mining engine with the task of sorting through the C4.5 algorithm using sampling data through the induction of decision trees. This algorithm also conducted an analysis of the data generating a classifier model, using statistical calculations to assess their results and see if it is generating correct information. Thus, the local generators uncertainties are excluded from the model resulting in an accuracy better than expected. The stage of implementation and testing of research carried out confirm the knowledge obtained during the study on the subject, which identifies the classification through the induction of decision trees as a process easily understood by humans, and to obtain satisfactory results in terms of time and level of accuracy in relation to other data mining tools.

Keywords: *Data Mining*; Classification; Decision Trees; C4.5 Algorithm; *Shell* Orion.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE SIGLAS

CART	*Classification and Regression Trees*
CHAID	*Chi-square Automatic Interaction Detector*
DM	*Data Mining*
ID3	*Induction of Decision Tree*
KDD	*Knowledge Discovery in Databases*
SLIQ	*Supervised Learning in Quest*
SQL	*Structure Query Language*
SVG	*Scalable Vector Graphics*
UML	*Unified Modeling Language*
UNESC	Universidade do Extremo Sul Catarinense

SUMÁRIO

1 INTRODUÇÃO

A utilização de bases de dados para armazenamento de informações tende a torná-las relativamente grande, sendo que alguns padrões contidos nelas ainda são desconhecidos, pois não foram processados de uma forma capaz de gerar conhecimento. A forma para explorar estes dados pode ocorrer por meio do processo de *Data Mining* (DM) que serve para extrair conhecimentos dos dados e descobrir relações significativas previamente desconhecidas (WITTEN; FRANK, 2005, tradução nossa).

O *data mining* é realizado por meio de tarefas como a classificação, associação, clusterização e previsão. Além das tarefas, são necessários métodos específicos para implementá-las, como por exemplo, as árvores de decisão, redes neurais artificiais, algoritmos genéticos.

Na utilização das tarefas e métodos se faz necessário o uso de ferramentas computacionais, sendo que, uma destas está em desenvolvimento por alunos e professores do Grupo de Pesquisa em Inteligência Computacional Aplicada do curso de Ciência da Computação da Universidade do Extremo Sul Catarinense. Esta ferramenta denomina-se *Shell* Orion *Data Mining* Engine, e possui implementadas as tarefas de associação, classificação e clusterização. No que se refere ao módulo da tarefa de classificação, o método primeiramente disponibilizado para indução das árvores de decisão foi o algoritmo ID3 (PELEGRIN, 2005), desenvolvido em meados de 1980 por John Ross Quinlan, o qual constrói uma árvore a partir de regras geradas após sua execução.

A evolução do código do ID3 gerou um outro algoritmo, o C4.5, também desenvolvido por John Ross Quinlan no ano de 1987, utilizado neste trabalho, com a mesma funcionalidade do primeiro, além de possuir a capacidade de simplificação da

árvore (poda), ou seja, exclui as regras que não têm valores significativos para a classificação, sendo possível desta maneira a realização da comparação dos resultados obtidos pelos dois algoritmos. A *Shell* Orion possui implementada além da tarefa de classificação, a associação pelo algoritmo *Apriori*, clusterização pelos algoritmos *K-means*, *Kohonen*, *Gustafson-Kessel e Gath-Geva*.

Objetivando-se apresentar diferentes resultados pela classificação na *Shell* Orion, esta pesquisa consistiu na implementação do algoritmo C4.5, pois este é a evolução do algoritmo ID3, sendo capaz de executar a simplificação da árvore o que resulta em uma classificação mais rápida e simples. Além disso, trabalha com atributos numéricos e nominais, com a possibilidade de classificar valores que estejam nulos no conjunto.

1.1 OBJETIVO GERAL

Desenvolver na tarefa de classificação da *Shell* Orion *Data Mining Engine* o método de árvores de decisão por meio do algoritmo C4.5.

1.2 OBJETIVOS ESPECÍFICOS

A seguir listam-se os objetivos específicos desta pesquisa:

a) compreender o processo de *data mining*, classificação e árvores de decisão;
b) compreender o algoritmo C4.5 para indução de árvores de decisão;
c) demonstrar matematicamente o funcionamento do algoritmo C4.5;
d) aplicar o algoritmo C4.5 para a tarefa de classificação em *data mining*;

e) utilizar uma base de dados a fim de testar as funcionalidades do algoritmo C4.5 implementado no módulo de classificação da *Shell* Orion.

1.3 JUSTIFICATIVA

A *Shell* Orion está em desenvolvimento pelo Grupo de Pesquisa em Inteligência Computacional Aplicada do curso de Ciência da Computação da UNESC, sendo que projetos como estes têm significativa importância no âmbito acadêmico e no mercado de trabalho, capacitando desta forma a disponibilização de uma ferramenta gratuita.

A realização de projetos como este também agrega conhecimento aos acadêmicos que, por sua vez, tem a capacidade de aprender e conhecer as visões de empreendimento e tomada de decisões, que são de grande importância para administradores e gerentes de negócio.

O intuito da realização desta pesquisa procura dar continuidade ao que já foi desenvolvido na *Shell* Orion, por meio da implementação do algoritmo C4.5 no módulo de classificação. Essa ação proporciona outras formas de se realizar esta tarefa, sendo que os resultados gerados na utilização de um algoritmo nem sempre são conclusivos quanto ao conhecimento obtido.

O algoritmo C4.5 foi implementado na *Shell* Orion pois é uma evolução do algoritmo ID3, uma vez que possui a capacidade de realizar a simplificação (poda) da árvore de decisão excluindo as regras que não apresentam valores significativos para a precisão da tarefa. Conforme Almentero, Baião e Mattoso (2004) e Motta (2004) esta

poda reduz a complexidade da árvore o que proporciona uma classificação mais rápida e eficiente.

Além de realizar a simplificação da árvore o algoritmo C4.5 também possui a capacidade de trabalhar com atributos numéricos e nominais, podendo estes conterem valores nulos no conjunto de dados analisado, o que permite acelerar o processo da descoberta de conhecimento, sendo que etapas como a de limpeza e a de transformação dos dados não necessariamente precisariam ser realizadas.

1.4 ESTRUTURA DO TRABALHO

Este trabalho é composto de sete capítulos, sendo o primeiro compreendido pela introdução do assunto estudado, apresentando também objetivos e justificativa da sua realização.

O Capítulo 2 disponibiliza conceitos da descoberta de conhecimento em bases de dados, que compreende dentre suas etapas fundamentos para o estudo do *data mining*.

No Capítulo 3 é realizada a explanação da tarefa de classificação e comentado sobre os métodos capazes de realizar tal feito. A execução da tarefa é empregada por meio de algoritmos, sendo apresentado os conceitos do algoritmo C4.5 no Capítulo 4.

Os Capítulos 5 e 6 são dedicados a demonstrar algumas das ferramentas de *data mining* que utilizam a tarefa de classificação pelo algoritmo C4.5 existentes no mercado de trabalho, bem como apresentar a *Shell* Orion, os seus métodos e a utilização destes.

O desenvolvimento do trabalho proposto e as etapas compreendidas na sua realização estão apresentadas no Capítulo 7, sendo seguido pela conclusão e algumas sugestões de projetos que poderão ser desenvolvidos juntamente a *Shell* Orion.

2 DESCOBERTA DE CONHECIMENTO EM BASES DE DADOS

A análise de grande volume de dados faz uso de processos do ramo da descoberta de conhecimento, como por exemplo o *data mining*, que visa extrair relações importantes entre informações contidas nas bases de dados. As diversas pesquisas que vêm sendo realizadas nesta área estão classificadas em *sequence data mining*[1], *web mining*[2], *text mining*[3] e *visual mining*[4] (TANIAR, 2008, tradução nossa).

O processo da descoberta consiste em tornar dados de baixo em alto nível para a demonstração do conhecimento ao usuário, no qual por meio de um grande volume de dados, tem-se o objetivo de extrair relações importantes de maneira eficiente e que possua informações relevantes (MITRA; ACHARYA, 2003, tradução nossa).

Há autores que tratam descoberta de conhecimento em bases de dados (*Knowledge Discovery in Databases*, KDD) como sendo o próprio *data mining*, e outros a descrevem como uma sequência de passos fundamentais para sua completa execução. Seguindo este conceito Han e Kamber (2001, tradução nossa) classificam as etapas do processo de KDD como:

a) **limpeza dos dados**: consiste na remoção de valores nulos, ilegais ou combinações inexistentes. Dados armazenados podem ser incompletos ou inconsistentes, não terem sido informados ou estarem armazenados com um valor divergente do esperado. Rotinas realizadas nesta etapa realizam a inclusão de valores previamente não informados, removem valores ilegais e corrigem as inconsistências dos dados;

---

[1] Baseia-se, por exemplo, no histórico do comportamento humano em relação as compras efetuadas no supermercado (DONG; PEI, 2007, tradução nossa).

[2] Compreende a mineração no histórico de acessos a *sites* na internet (DONG; PEI, 2007, tradução nossa).

[3] Técnica de data mining especial para busca de informações em bases de dados de publicações eletrônicas, bibliotecas digitais, e-mail, entre outros (KANTARDZIC, 2003, tradução nossa).

[4] Refere-se a análise de imagens e busca de informações nas mesmas, sendo mais utilizado em estudos geográficos (TANIAR, 2008, tradução nossa).

b) **integração dos dados:** na existência de várias bases de dados, realiza-se uma combinação entre elas. Deve-se ter muito cuidado na integração das bases, pois esta etapa pode resultar na inconsistência e na redundância dos dados;

c) **seleção dos dados:** uma base de dados contendo uma quantidade de informações muito grande pode aumentar consideravelmente o tempo de execução do *data mining.* Nesta etapa, então, realiza-se a escolha aleatória de uma quantidade de registros da base com a finalidade de minimizar o tempo de processamento. Assim, obtém-se um resultado idêntico ou parecido com o que seria gerado utilizando-se a base completa;

d) **transformação dos dados:** etapa que ocorre no caso da necessidade de alguma transformação dos dados a fim de prepará-los para a execução do *data mining*. Como exemplo, tem-se a conversão de dados nominais[5] para numéricos na aplicação do algoritmo *K-means*[6] na tarefa de clusterização, pois este trabalha somente com atributos numéricos;

e) ***data mining*:** realiza a exploração e análise dos dados em busca da descoberta de uma regra ou relação que possua uma importância para a tomada de decisões (BERRY; LINOFF, 2004, tradução nossa);

f) **avaliação das relações:** identificam-se as relações obtidas na etapa anterior, avaliando se estas representam o conhecimento baseado no fundamento do assunto. Apenas os resultados de interesse do usuário devem ser utilizados;

---

[5] Atributos não-numéricos que possuem um número finito (mas possivelmente grande) de valores distintos (HAN; KAMBER, 2001, tradução nossa).

[6] Algoritmo de particionamento utilizado na tarefa de clusterização que visa agrupar conjuntos de dados por meio do critério de erro encontrado na distância dos dados (KANTARDZIC, 2003, tradução nossa).

g) **apresentação do conhecimento:** etapa final, quando os resultados obtidos no *data mining* são apresentados ao usuário, podendo variar no modelo de visualização escolhido, como por exemplo, regras, tabelas, gráficos, árvores de decisão, entre outros.

A Figura 1 demonstra a sequência das etapas realizadas no KDD como comentado.

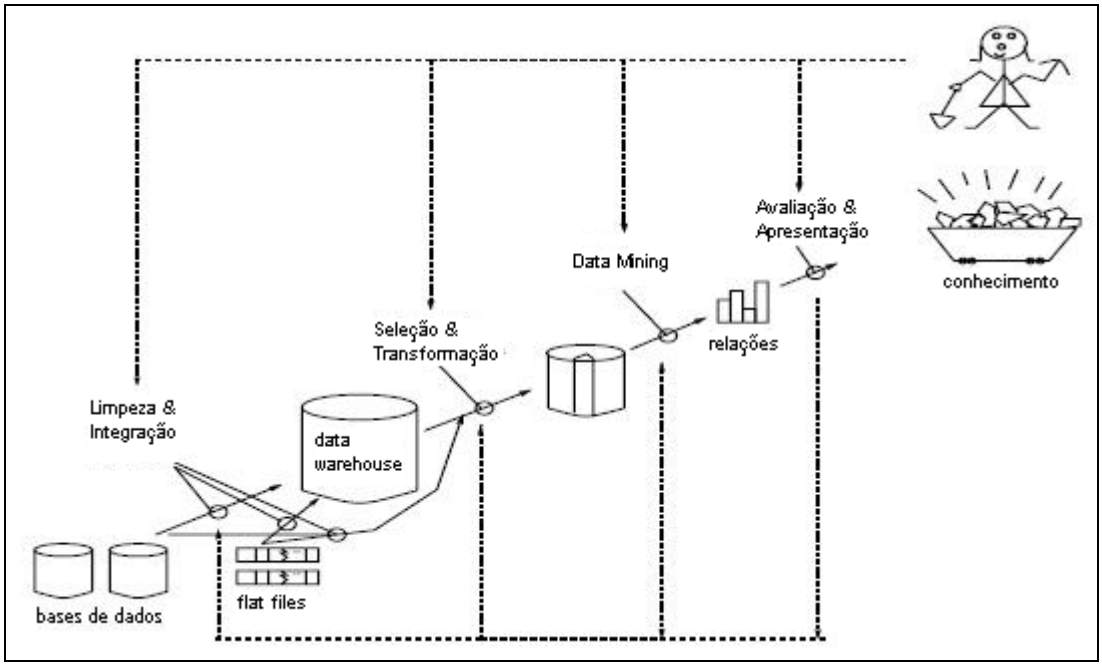

Figura 1. Etapas do processo de KDD
Fonte: Traduzido de HAN, J.; KAMBER, M. (2001, tradução nossa)

Dentre as etapas realizadas no KDD, a que possui maior importância para este trabalho é a de *data mining*, pois é nela que executam-se os algoritmos das diferentes tarefas disponíveis, proporcionando a obtenção dos resultados.

Estes resultados são os padrões e as relações descobertas que podem confirmar o que já se conhece ou apresentar conhecimentos novos.

2.1 *DATA MINING*

O constante avanço da tecnologia de informação tem contribuído com o crescimento e surgimento de bases de dados responsáveis pelo armazenamento de informações em grandes proporções. Como exemplo pode-se citar: missões da NASA, Projeto do Genoma Humano, instituições como FedEx, Banco do Brasil e Caixa Econômica Federal, onde suas bases de dados contém centenas de terabytes de informação (GOLDSCHMIDT; PASSOS, 2005).

A necessidade de entender estas bases repletas de informações é comum nas mais diversas áreas de atuação, tais como: comércio, ciência, saúde, engenharia, entre outros. A habilidade de extrair de maneira proveitosa o conhecimento dessas bases de dados atrai cada vez mais a atenção das instituições, como recurso estratégico no competitivo mercado de trabalho (KANTARDZIC, 2003, tradução nossa).

Diante de casos como estes surgem questionamentos sobre o que fazer com os dados armazenados, como analisá-los e utilizá-los de maneira útil. Em virtude da grande quantidade é inviável realizar a análise destes dados, portanto, torna-se necessária a utilização de ferramentas computacionais apropriadas (GOLDSCHMIDT; PASSOS, 2005).

O processo de inclusão de novas técnicas de busca de conhecimento em dados com a utilização de computadores é chamado de *data mining* (KANTARDZIC, 2003, tradução nossa).

Data Mining pode ser definido como a extração de novas, e desconhecidas, informações de bases de dados, ou ainda ser apenas a confirmação de fatos já conhecidos. Sua aplicação tem principal interesse em descobrir tendências inesperadas e relações não tão óbvias entre os dados contidos nas bases, onde todo o processo envolve

técnicas de aprendizado de máquina, estatística e noções de bancos de dados (TANIAR, 2008, tradução nossa).

A execução do *data mining* é realizada por meio de técnicas de busca e de amostragem sobre os dados, constituindo-se em uma ferramenta de auxílio para explicação das informações e de possíveis previsões sobre relações que possam vir a ocorrer com frequência (WITTEN; FRANK, 2005, tradução nossa).

Data mining (DM) é um processo iterativo definido pelo descobrimento de conhecimento em dados por métodos tanto automáticos quanto manuais. Portanto, é a cooperação do trabalho humano em conjunto com o computador, destacando-se em cenários de análise exploratória no qual não há noções pré-determinadas sobre o resultado a ser obtido (KANTARDZIC, 2003, tradução nossa).

Assim, o DM compreende a identificação de relações entre os dados possibilitando a descoberta de conhecimento. Esta descoberta pode ser realizada de diferentes formas, de acordo com o seu objetivo, pela aplicação das tarefas de DM.

2.1.1 Tarefas de *data mining*

As tarefas de *data mining* são modelos de resolução de problemas, que diversas técnicas tem a capacidade de descobrir relações entre as informações contidas em uma base de dados (CUNICO, 2005). As tarefas utilizadas, por sua vez, são realizadas por diferentes algoritmos. Dentre elas tem-se: classificação, estimativa, previsão, associação e clusterização.

2.1.1.1 Classificação

Consiste em examinar as características de objetos apresentados e designá-los a classe de um grupo pré-definido. Os objetos a serem classificados são geralmente representados por registros de tabelas das bases de dados, sendo que a ação de classificação tende a incluir um novo valor na classe (BERRY; LINOFF, 2004, tradução nossa).

Caracteriza-se pelo processo de busca de modelos ou funções que possam descrever e distinguir relações ou conceitos entre os dados, com o propósito de prever características entre informações ainda não armazenadas na base. Estes modelos podem ser representados de diferentes maneiras, como por exemplo, regras de classificação[7], árvores de decisão[8] ou redes neurais[9] (HAN; KAMBER, 2001, tradução nossa).

A tarefa de classificação pode ser empregada a fim de se obter, por exemplo (BERRY; LINOFF, 2004, tradução nossa):

a) liberação de crédito para clientes de um banco, sendo estes classificados de baixo, médio ou alto risco;
b) classificação de conteúdos a serem mostrados em uma página na web.

Na Figura 2 pode-se visualizar a tarefa da classificação utilizada para separar em duas classes os símbolos gráficos apresentados.

---

[7] Blocos de comandos na forma SE – ENTÃO, que facilitam a compreensão pelo usuário (HAN; KAMBER, 2001, tradução nossa).

[8] Estrutura na forma de uma árvore, que se comparada as regras de classificação, tem-se os nós identificados como um comando SE, enquanto as folhas denotam o ENTÃO (HAN; KAMBER, 2001, tradução nossa).

[9] Estrutura representada por conexões de entrada e saída para classificar casos conforme os pesos sinápticos (HAN; KAMBER, 2001, tradução nossa).

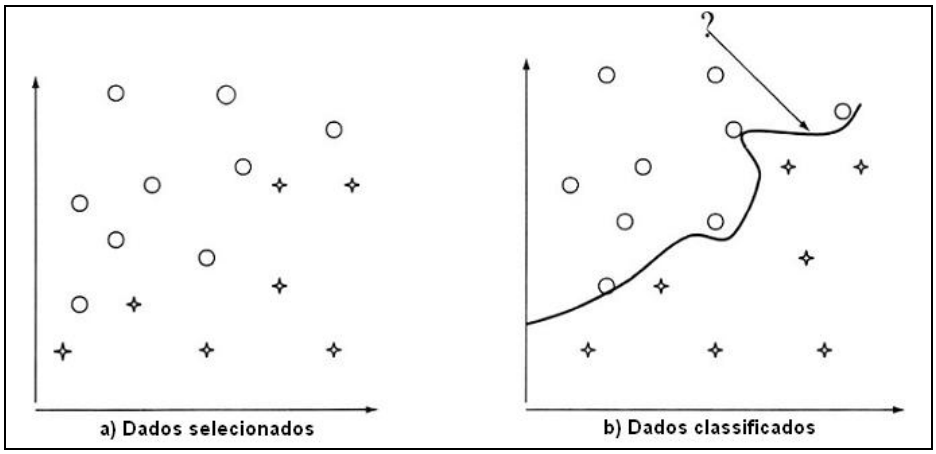

Figura 2. Interpretação gráfica da classificação
Fonte: Traduzido de KANTARDZIC, M.(2003, tradução nossa)

2.1.1.2 Estimativa

A estimativa é uma tarefa similar a classificação, exceto pelo fato de que a variável analisada é do tipo numérica. A execução da tarefa se utiliza de todos os atributos de um registro que venha a prover informação, para estimar um resultado desejado, sendo realizado pela observação dos valores existentes nos atributos além do estimado. Por exemplo, em um hospital para estimar qual deve ser a pressão sistólica do sangue do paciente baseia-se na idade, sexo, índice de massa corpórea e nível de sódio no sangue. A relação entre a pressão sistólica do sangue e os fatores levantados, pode ser utilizada como um modelo de estimativa, para assim aplicá-lo a novos casos (LAROSE, 2005, tradução nossa).

Alguns exemplos desta tarefa consistem em estimar (SOUKUP; DAVIDSON, 2002, tradução nossa):

a) valor gasto por um cliente, anualmente, em uma empresa de cartão de crédito;

b) registros de reclamações ao fabricante de um produto;

c) lucro gerado pelo cliente de uma empresa de telecomunicações.

2.1.1.3 Previsão

Constrói um ou mais modelos para avaliar os dados contidos na base, podendo assim prever o comportamento de registros que ainda serão inseridos nela. A tarefa consiste também em determinar valores de atributos que estejam nulos, no caso de registros não descartados para a realização de outras tarefas do *data mining* (FAN, 2004, tradução nossa).

Previsão, assim como a estimativa, é similar a classificação, sendo que nesta tarefa são determinados resultados futuros. Na tarefa de previsão, apenas após o acontecimento do fato pode-se verificar se o resultado obtido foi exato ou não (BERRY; LINOFF, 2004, tradução nossa).

Segundo Larose (2005, tradução nossa), alguns exemplos da utilização da tarefa é prever:

a) o preço de um produto no estoque daqui a três meses;
b) o aumento na porcentagem de mortes nas rodovias para o próximo ano se o limite de velocidade for aumentado;
c) o vencedor do campeonato brasileiro de futebol, baseado nas estatísticas de cada time.

2.1.1.4 Associação

A tarefa de associação visa descobrir quais atributos devem ficar juntos. Conhecida como análise de afinidade, a tarefa de associação tende a descobrir regras para qualificar a relação entre dois ou mais atributos. Regras de associação estão na

forma SE - ENTÃO, juntamente com medidas de suporte[10] e confiança[11], que são demonstradas por valores de confirmação dos registros associados. Por exemplo, um determinado supermercado pode descobrir que de 1000 clientes que compraram em uma quinta a noite, 200 compraram fraldas, e que destes 200, 50 compraram cerveja. Sendo assim, a regra de associação seria "Se compra fraldas então compra cerveja" com o suporte de 200/1000 = 20% e a confiança de 50/200 = 25% (LAROSE, 2005, tradução nossa).

A descoberta de consideráveis relações de associação contidas nas bases pode auxiliar nas decisões de marketing, e na análise de gerentes sobre vendas dos processos internos da empresa (HAN; KAMBER, 2001, tradução nossa).

Além destes, tem-se também como exemplos da tarefa (LAROSE, 2005, tradução nossa):

a) examinar a proporção de crianças cujos pais leram para elas, que se tornaram bons leitores;
b) determinar a proporção de casos em que uma nova droga exibe graves efeitos colaterais;
c) investigar a proporção de assinantes de um plano de uma companhia de celular que respondeu positivamente a uma oferta de melhoria de um serviço.

[10] Refere-se a proporção de registros do conjunto de dados presentes num determinado nó da árvore ou de uma regra analisada (LAROSE, 2005, tradução nossa).

[11] Especifica a proporção dos registros corretamente classificados referindo-se a regra ou nó analisado (LAROSE, 2005, tradução nossa).

2.1.1.5 Clusterização

A tarefa de clusterização divide os dados em um número menor de subgrupos ou *clusters* (grupos). A meta da tarefa é descobrir similaridades previamente desconhecidas entre os dados. A clusterização é um ótimo meio de iniciar a análise sobre os dados, pois pode definir pontos de partida para a descoberta de relações entre subgrupos (RUD, 2001, tradução nossa).

Alguns exemplos da utilização da tarefa de clusterização são (LAROSE, 2005, tradução nossa):

a) auditoria de contas bancárias para segmentar o comportamento financeiro em categorias de bom ou mau intencionados;
b) utilizar como ferramenta para reduzir a dimensão de bases de dados com centenas de atributos;
c) realizar a clusterização de expressão de genes, onde grande quantidade de genes podem exibir comportamentos similares.

Os resultados originados por este método visam a amostragem de conjuntos de dados de devida importância, sem um resultado específico. Este método é útil em bases de dados com centenas de atributos a serem avaliados, para assim poder realizar uma busca de conhecimento nos conjuntos obtidos (REA, 2005, tradução nossa).

A Tabela 1 apresenta de forma simplificada o que se pode obter com a utilização das tarefas anteriormente descritas.

Tabela 1. Exemplos das tarefas de *data mining*

Tarefa	Objetivo
Classificação	Realizar a análise em dados com o objetivo de classificá-los a um determinado tipo de classe
Estimativa	Analisar relação entre atributos que são geradores de um determinado atributo, conhecendo assim o que poderá vir a ocorrer em função de novos dados inseridos na base
Previsão	Avaliar o comportamento dos dados em virtude de conhecer o que virá a ser inserido na base
Associação	Buscar afinidade entre atributos da base de dados analisada a fim de identificar quais dependem uns dos outros
Clusterização	Agrupar dados em grupos menores para identificar possíveis relações entre os subgrupos gerados

Dentre as tarefas de DM apresentadas, a classificação foi empregada neste trabalho objetivando-se a construção de um modelo classificador interpretável pelo usuário.

3 A TAREFA DE CLASSIFICAÇÃO

Esta tarefa tem como objetivo realizar uma função de classificação dos dados analisados a fim de mapeá-los com suas respectivas classes. Um classificador realiza a função a partir de um grupo de dados mapeando as suas respectivas classes, podendo assim classificar novos registros. A classificação é dita como um aprendizado supervisionado[12], pois os registros possuem suas classes pré-definidas a partir do grupo inicialmente analisado (TANIAR, 2008, tradução nossa).

Considerada uma das mais populares tarefas de *data mining*, ela é aplicada, por exemplo, em problemas comerciais como: análise de produções em série e controle gerencial dos processos da empresa (TANG; MACLENNAN, 2005, tradução nossa).

A tarefa citada é definida como a busca por uma função que classifique um registro de uma base de dados a um rótulo categórico, denominado classe. Por meio desta, novos registros podem ser classificados de acordo com suas características, conforme está apresentado na Figura 3 (GOLDSCHMIDT; PASSOS, 2005).

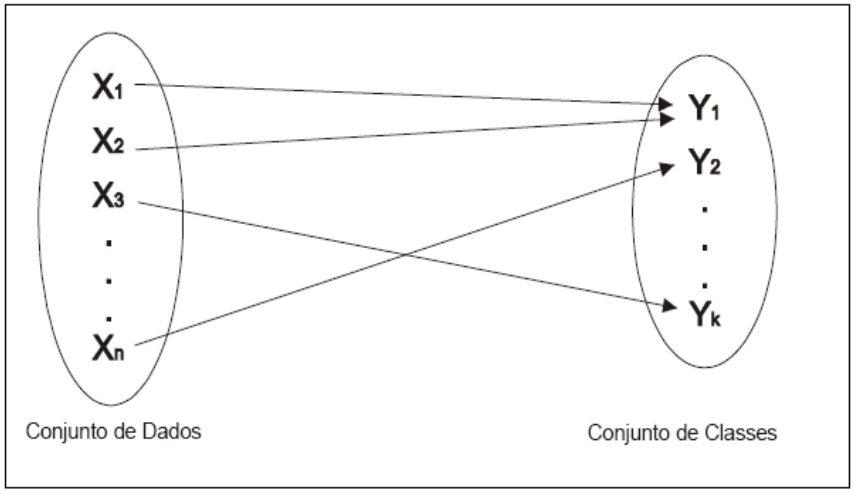

Figura 3. Relação entre registros de dados e classes
Fonte: GOLDSCHMIDT, R.; PASSOS, E.(2005).

[12] O aprendizado se diz ser supervisionado, pois já está designado a qual classe um determinado registro pertence (HAN; KAMBER, 2001, tradução nossa).

Han e Kamber (2001, tradução nossa) definem que a tarefa é um processo constituído de dois passos: a construção do modelo de classificação e a avaliação dos testes realizados. No primeiro passo, um modelo é construído descrevendo classes ou conceitos de dados pré-selecionados, baseando-se em tuplas[13] da base de dados descritas por seus atributos, podendo também se referenciar a exemplos, amostras ou objetos. Cada tupla deverá pertencer a uma classe em particular, determinada por um de seus atributos, chamado de atributo rótulo da classe. As tuplas analisadas para construir o modelo formam o conjunto de instrução dos dados. Cada uma refere-se a uma amostra da instrução, que são selecionadas aleatoriamente do conjunto de amostras.

No segundo passo, o modelo é utilizado para realizar a classificação, obtendo-se o nível de exatidão da execução da tarefa, que corresponde a porcentagem dos testes realizados que foram corretamente classificados (HAN; KAMBER, 2001, tradução nossa).

Caso a exatidão do modelo seja considerada aceitável, este pode ser utilizado para a classificação de tuplas futuras ou objetos que ainda não foram classificados (HAN; KAMBER, 2001, tradução nossa).

Geralmente, o modelo de aprendizagem é apresentado na forma de regras de classificação, árvores de decisão ou equações matemáticas. Por exemplo, tendo-se uma base de dados com informações de crédito de clientes, regras de classificação podem avaliar clientes como sendo bons ou excelentes em razão do crédito (HAN; KAMBER, 2001, tradução nossa).

Na Figura 4 pode-se ver a base de dados sendo representada com informações sobre os clientes, onde por meio de um algoritmo e de seu resultado

[13] Lista de valores representada por seus atributos, denotando uma linha da tabela do banco de dados (GARCIA-MOLINA; ULLMAN; WIDOM, 2001).

gerado, é possível ser classificado uma nova tupla como demonstra a Figura 5, que classifica o cliente como excelente para a obtenção de crédito.

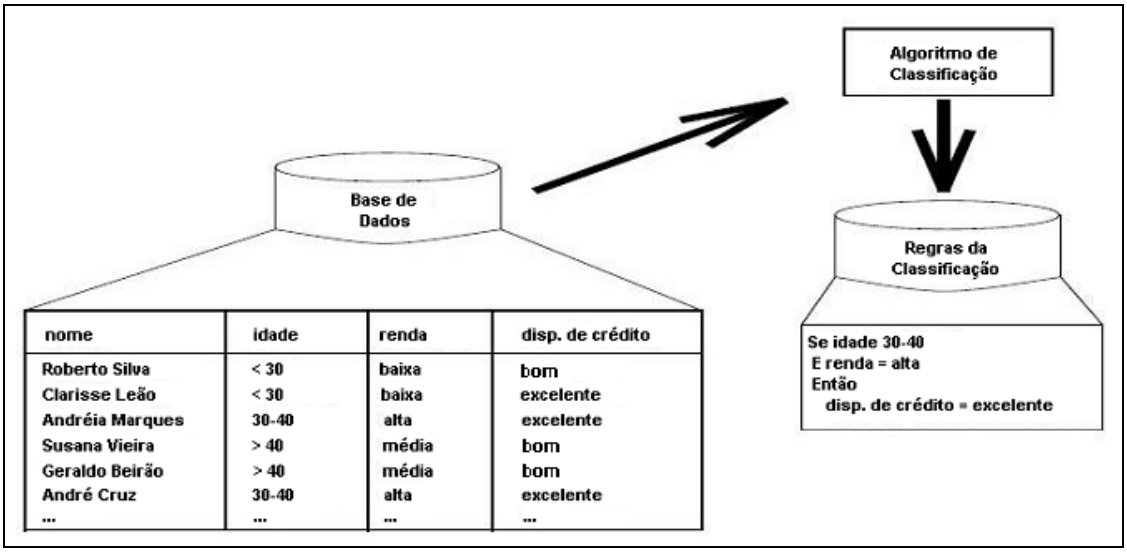

Figura 4. Classificação para geração das regras
Fonte: Traduzido de HAN, J.; KAMBER, M. (2001, tradução nossa)

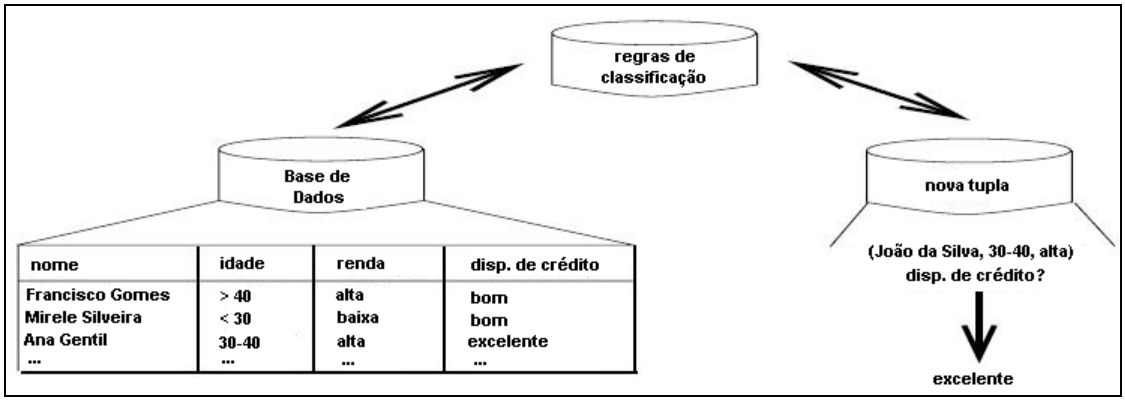

Figura 5. Classificação por meio das regras geradas
Fonte: Traduzido de HAN, J.; KAMBER, M. (2001, tradução nossa)

Na tarefa de classificação para a construção do modelo utilizam-se alguns métodos de DM, tais como: redes neurais artificiais, algoritmos genéticos, árvores de decisão, entre outros.

3.1 MÉTODOS UTILIZADOS NA TAREFA DE CLASSIFICAÇÃO

Os métodos utilizados para a criação de um modelo classificador têm como principal função, explorar a capacidade de tomada de decisões em um ambiente composto de dados imprecisos e também incertos (MITRA; ACHARYA, 2003, tradução nossa). Dentre os métodos utilizados na tarefa de classificação, tem-se:

a) **redes neurais artificiais:** o cérebro é composto por milhões de neurônios, que funcionam como unidades de processamento. Estes são considerados a fonte da capacidade de aprendizado e inteligência do ser humano. De maneira semelhante às funções realizadas pelos neurônios, as redes neurais artificiais são implementadas na forma computacional visando simular aspectos de rapidez, adaptação e aprendizado sobre as informações processadas. Assim, este método visa descobrir automaticamente relações entre os dados do conjunto analisado (MITRA; ACHARYA, 2003, tradução nossa). O modelo de uma rede neural pode ser demonstrado pela Figura 6, onde (LAROSE, 2005, tradução nossa):

   - *Node* 1, *Node* 2 e *Node* 3 representam a camada de entrada, ou seja, o conjunto de dados a ser analisado;
   - *Node* A e *Node* B a camada oculta, podendo ser interpretado como os dados de entrada com algumas abstrações em suas características;
   - e *Node* Z, representando a camada de saída.

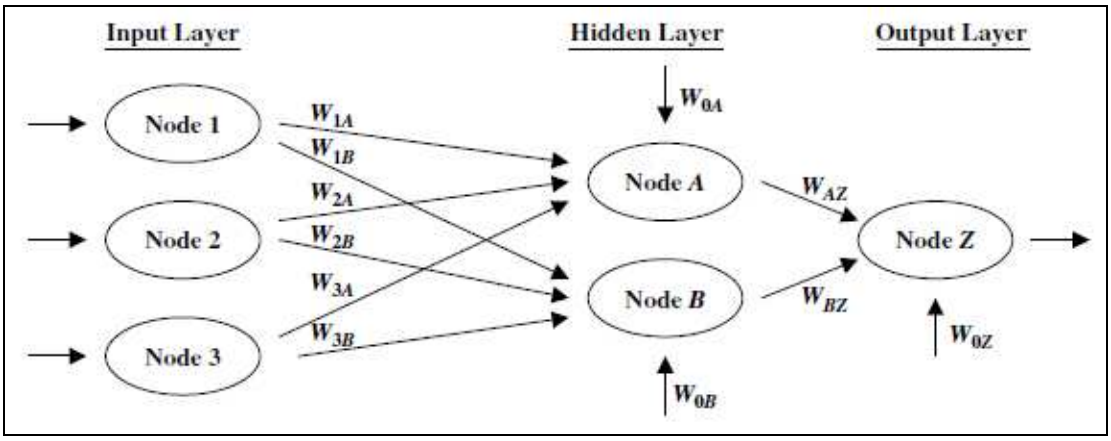

Figura 6. Exemplo de rede neural artificial
Fonte: Extraído de LAROSE, D. T. (2005)

b) **algoritmos genéticos:** os algoritmos genéticos têm o objetivo de se comportar conforme um quadro de soluções evolutivas que operam na transferência de genes, onde a partir deste quadro tem-se a necessidade de interpretar qual das soluções contribuirá mais para as gerações futuras, agindo de forma adaptativa com o conjunto de dados. A Figura 7 demonstra a transição dos genes durante um quadro evolutivo, onde cadeias de genes tendem a sumir e outras a se multiplicar (LUGER, 2004);

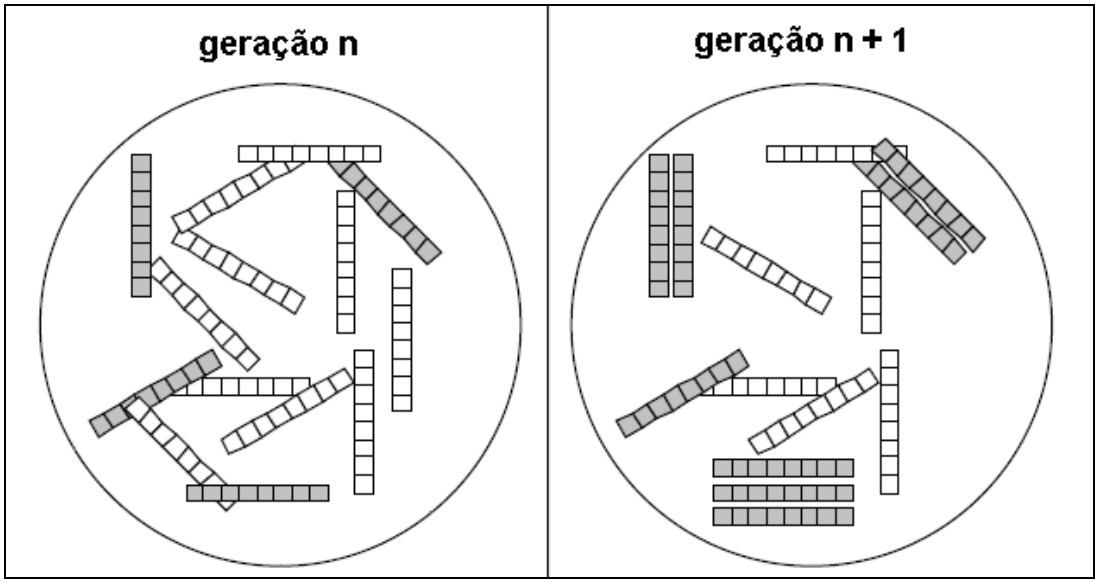

Figura 7. Exemplo da evolução por meio de um algoritmo genético
Fonte: Adaptado de BERRY, M. J. A.; LINOFF, G. (2004, tradução nossa)

c) **redes bayesianas:** a classificação é realizada neste método por meio de cálculos probabilísticos baseados no Teorema de Bayes, examinando registros e atributos de uma base de dados como forma de busca de relações entre os valores. A análise é realizada considerando a ocorrência de valores de um atributo que possam ter ligação com o valor de outro atributo, sendo capaz de realizar classificações baseadas na probabilidade encontrada entre eles e antecessores ligados ao primeiro atributo, assim como pode ser visualizado na Figura 8, onde a probabilidade de valores de um atributo depende de outro (MITRA; ACHARYA, 2003, tradução nossa);

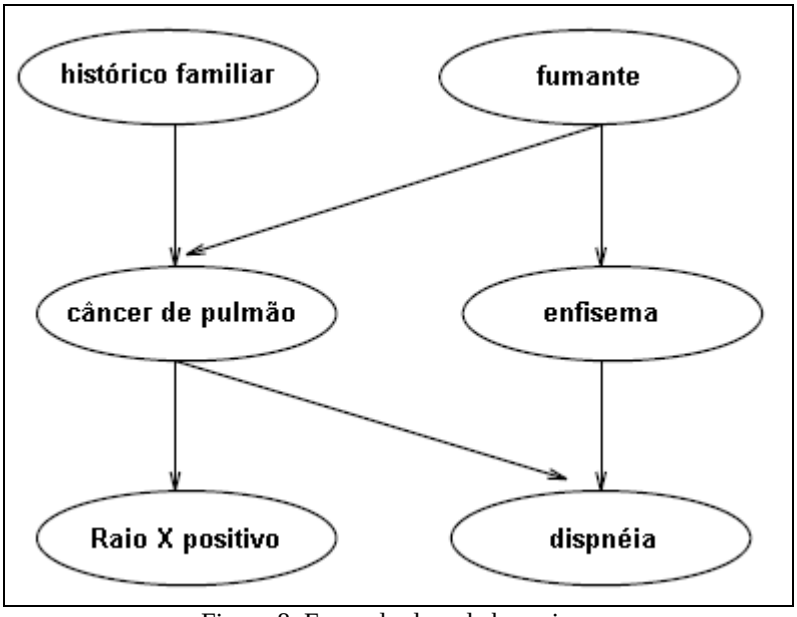

Figura 8. Exemplo de rede bayesiana
Fonte: Traduzido de HAN, J.; KAMBER, M. (2001, tradução nossa)

d) **árvores de decisão:** são demonstradas no modelo de árvore iniciando a partir da raiz e indo em direção a uma de suas folhas. Um caso a ser classificado percorre a árvore passando por seus nós de decisão, onde os aspectos encontrados no registro determinam o caminho a ser traçado. Ao encontrar uma das folhas da árvore o registro é atribuído a classe

informada. A Figura 9 representa uma árvore construída com sua raiz no topo, nós intermediários e folhas nas extremidades (QUINLAN, 1993, tradução nossa).

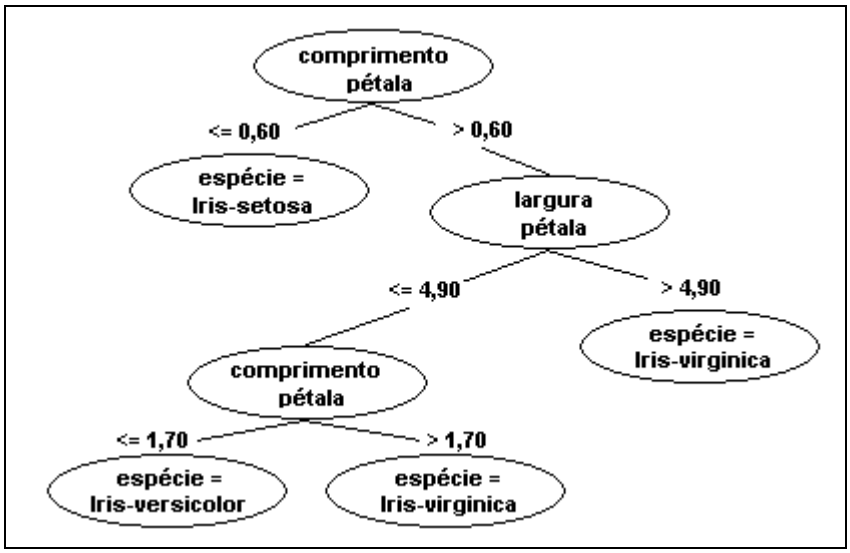

Figura 9. Exemplo de árvore de decisão

Dentre os métodos de classificação apresentados existem suas vantagens e desvantagens, dispostas na Tabela 2.

Tabela 2. Métodos empregados na tarefa de classificação

MÉTODOS	VANTAGENS	DESVANTAGENS
Redes Neurais	Capacidade de adquirir conhecimento automático a partir dos dados do conjunto	Complexidade da compreensão na demonstração dos resultados obtidos
Algoritmos Genéticos	Alta capacidade de adaptação aos registros	Tempo de processamento na demonstração de resultados em grande escala de dados
Redes Bayesianas	Simplicidade na interpretação das questões analisadas	Tempo prolongado de processamento em função dos cálculos utilizados
Árvores de Decisão	Fácil demonstração e entendimento lógico dos casos	Utilização de muitos atributos e muitos registros resultam em uma quantidade de regras de difícil compreensão

Fonte: GOLDSCHMIDT, R.; PASSOS, E. (2005); LUGER, G. F. (2004); MITRA, S.; ACHARYA, T. (2003, tradução nossa).

O resultado gerado pela classificação pode ser demonstrado na forma de árvore de decisão, onde cada atributo é figurado como uma ramificação da árvore. Neste trabalho, dentre os métodos de classificação, aplica-se o de árvores de decisão.

3.2 O MÉTODO DE ÁRVORES DE DECISÃO

Ao aplicar-se o método de árvores de decisão a uma base de dados tem-se a formação de uma árvore que possibilita ao usuário traçar um caminho específico na conquista de um resultado satisfatório (REA, 2005, tradução nossa).

Esta é uma estrutura de visualização de regras parecida com as ramificações de uma árvore, sendo que (HAN; KAMBER, 2001, tradução nossa):

a) **nó:** representa o teste feito ao valor de um atributo;

b) **ramificação:** é o resultado do teste ao nó;

c) **folha:** consiste na distribuição das classes.

As árvores de decisão podem ser facilmente convertidas e reapresentadas no modelo de regras de classificação (HAN; KAMBER, 2001, tradução nossa).

A maioria dos métodos de criação de árvores de decisão utiliza uma abordagem raiz – folha, *top - down*, questionando-se qual atributo deve ser usado para definir uma partição. Cada atributo é avaliado por meio do ganho de informação[14], para determinar qual deles separadamente pode classificar melhor um conjunto de dados. O processo é então repetido utilizando as amostras associadas a cada nó descendente, para selecionar o melhor atributo a representar uma nova partição, caso esta exista. As partições são como uma grade de busca para a árvore, em que o algoritmo não é capaz de retornar um nó para reconsiderar escolhas anteriores (YE, 2003, tradução nossa).

Uma árvore de decisão pode ser usada para dividir grandes quantidades de registros em conjuntos menores aplicando uma regra de decisão. A cada divisão sucessiva, os membros dos conjuntos resultantes tornam-se mais similares uns aos outros. Uma divisão familiar é a dos seres vivos, que são classificados em reino, fila,

[14] Cálculos estatísticos baseados na teoria da informação, também conhecido como entropia (YE, 2003, tradução nossa). Etapas e exemplos estão definidos no Capítulo 4.

classe, ordem, família, gênero e espécie. O modelo de uma árvore de decisão consiste no conjunto de regras que divide dados heterogêneos em grupos mais homogêneos, respeitando um atributo em particular (BERRY; LINOFF, 2004, tradução nossa).

Árvores de decisão são de fácil interpretação, pois podem ser representadas na forma de regras SE - ENTÃO. Na exploração de dados, as árvores de decisão contribuem, apresentando como vantagens (MITRA; ACHARYA, 2003, tradução nossa):

a) redução do volume de dados transformando-o em uma forma mais compacta, preservando as características essenciais;
b) capacidade de descobrir se os dados possuem classes de objetos bem separadas, para que assim, as classes possam ser interpretadas corretamente no contexto da classificação;
c) transformação de um processo de decisão complexo em uma coleção de decisões mais simples, providenciando assim uma solução de fácil interpretação.

Algoritmos de indução de árvores de decisão vêm sendo utilizados para classificação em variados tipos de aplicações, pois os passos de aprendizado e classificação induzidos por este método são geralmente rápidos, além de apresentarem uma exatidão tipicamente alta (HAN; KAMBER, 2001, tradução nossa).

Algumas das desvantagens encontradas na construção das árvores de decisão são (ROKACH; MAIMON, 2008, tradução nossa):

a) a maioria dos algoritmos necessitam que o atributo de saída seja do tipo nominal, tendo-se como exemplo os algoritmos ID3[15] e C4.5[16];

[15] Algoritmo para indução de árvores de decisão por meio da análise de relações no conjunto de dados (QUINLAN, 1993, tradução nossa).
[16] Evolução do algoritmo ID3, sendo capaz de analisar o modelo classificador construído, a fim de reduzir a taxa de erro encontrada, para isso realiza a poda da árvore (QUINLAN, 1993, tradução nossa).

b) sensibilidade em relação a base de dados no que se refere a atributos irrelevantes e distúrbios nos dados, deixando assim as árvores geradas muito instáveis, pois a menor alteração num valor de divisão próximo a raiz irá alterar toda a estrutura da árvore anteriormente gerada. Resultando, desta forma, na escolha de um atributo que não seja realmente o melhor para tal divisão.

A fim de evitar esta inconsistência, existe a poda das árvores, que pode ser realizada de duas maneiras segundo Han e Kamber (2001, tradução nossa):

a) **pré-poda**: este caso ocorre durante a construção da árvore, mais especificamente, na avaliação da realização de uma partição em um determinado nó. O resultado obtido pelo ganho de informação do atributo deve satisfazer um limiar pré-especificado, caso isto não ocorra, a partição não é realizada no nó avaliado;

b) **pós-poda**: remove ramificações de uma árvore já construída. Nesta etapa se utiliza um algoritmo de poda de árvores, por exemplo, complexidade do erro[17] ou erro pessimista[18]. Inicialmente, para cada nó não-folha da árvore, é calculada a taxa de erro[19] esperada, caso sua sub-árvore seja podada. Em seguida, calcula-se a taxa de erro para cada ramificação caso a sub-árvore não seja podada, combinada com a importância de cada ramificação. Se a poda do nó resultar na taxa de erro acima do esperado, então a sub-árvore é mantida, caso contrário, ela é podada. Após a geração de sucessivas podas na árvore, um conjunto de

[17] Método para cálculo dos resultados classificados erroneamente, encontradas na realização da poda em um nó de decisão (HAN; KAMBER, 2001, tradução nossa).

[18] Método de avaliação de resultados corretamente classificados que utiliza uma parte do conjunto de dados não agregada na construção do modelo classificador (HAN; KAMBER, 2001, tradução nossa).

[19] Valor obtido na avaliação da classificação de um determinado registro (HAN; KAMBER, 2001, tradução nossa)

dados independentes é utilizado para calcular a exatidão de cada modelo da árvore podada. Aquela que possuir a menor taxa de erro será escolhida como modelo final.

De acordo com Wang e Fu (2005, tradução nossa) o conhecimento aprendido pelas árvores de decisão pode ser demonstrado facilmente na forma de regras, porém, está propenso a erros no caso de haver distúrbio nos dados.

A fim de compreender o método de indução de árvores de decisão na tarefa de classificação se faz necessário um estudo sobre os algoritmos empregados na sua execução.

3.3 ALGORITMOS DE CLASSIFICAÇÃO PARA INDUÇÃO DE ÁRVORES DE DECISÃO

A tarefa de classificação tem como objetivo analisar um registro e atribuí-lo a uma classe, baseando-se na descoberta de padrões observados em uma base de dados. Como exemplo tem-se: análise médica referente a doença de seu paciente; busca por uma flor vermelha em imagens numa base de imagens; consulta de documentos referenciando o assunto *data mining* em um repositório. Dentre estes exemplos citados, os algoritmos que utilizam o método de indução de árvores de decisão podem estar divididos como (MITRA; ACHARYA, 2003, tradução nossa):

a) **classificadores para imensas bases de dados:** tem-se os algoritmos SLIQ, SPRINT e RainForest, entre outros. A utilização destes algoritmos visa melhorar a performance na sua execução, pois já que uma grande quantidade de dados está contida na base, estes realizam o

processamento de forma diferenciada, podendo utilizar mais de um processador ou mais de uma máquina;

b) **classificadores de aprendizado de máquina:** CHAID, CART, ID3 e C4.5, entre outros. Estes, por sua vez, são algoritmos que tratam os dados a fim de simular um comportamento humano, por exemplo, a tarefa de classificar pessoas por suas características físicas aparentes. Assim, analisando-se diferentes aspectos nos dados, pode-se compreender e gerar resultados similares ao raciocínio lógico empregado pelo cérebro humano.

3.3.1 Supervised Learning in Quest (SLIQ)

A publicação do algoritmo SLIQ foi realizada em 1996 por Manish Mehta, Rakesh Agrawal e Jorma Rissanen. O SLIQ é um classificador que utiliza a indução de árvores de decisão e trabalha com atributos numéricos e nominais.

A construção da árvore se torna mais rápida devido ao fato deste algoritmo pré-ordenar os dados da base, sendo que outros utilizam a re-ordenação a cada nó em análise. O SLIQ, por meio de um algoritmo de poda de árvore é capaz de modelar árvores compactas e precisas (APERS; BOUZEGHOUB; GARDARIN, 1996, tradução nossa).

3.3.2 SPRINT

Este é a evolução do algoritmo SLIQ, tendo sido publicado no mesmo ano, 1996, por John C. Shafer, Rakesh Agrawal e Manish Mehta. O SPRINT tem como foco a execução paralela entre mais de um processador.

O algoritmo é capaz de realizar o processamento compartilhado com outras máquinas, não usando apenas a memória local para alocação dos dados contidos na base, usufruindo desta forma dos componentes de armazenamento e processamento compartilhados entre elas (HELLERSTEIN; STONEBRAKER, 2005, tradução nossa).

3.3.3 RAINFOREST

Publicado em 2000 por Johannes Gehrke, Raghu Ramakrishnan e Venkatesh Ganti, é uma extensão de um conjunto de vários algoritmos.

O armazenamento dos valores dos atributos é feito por um método chamado *Attribute Value, Class – set* (AVC-set), uma forma compacta de analisar os dados onde cada possível valor diferente é alocado na memória em vez de alocar todos os registros contidos na base de dados (PUJARI, 2001, tradução nossa). A construção da árvore e os processos executados neste algoritmo são uma junção de técnicas encontradas nos algoritmos: C4.5, CART, CHAID E SPRINT (MITRA; ACHARYA, 2003, tradução nossa).

3.3.4 CHAID

Em 1975 foi publicado por John A. Hartigan o algoritmo *Chi-square Automatic Interaction Detector* (CHAID) que detecta relações estatísticas entre atributos de uma base de dados. O algoritmo faz uso do teste *chi-square* que mede a probabilidade de uma diferença entre amostras ser meramente casual, ao realizar as seguintes etapas (BERRY; LINOFF, 2004, tradução nossa):

a) na união de classes que não possuem diferença significativa sobre o *atributo rotulador*[20];
b) ao escolher a melhor partição para o nó;
c) na verificação da possibilidade de ser realizada mais alguma partição adicional em um dos nós gerados.

O algoritmo original aplica o teste *chi-square* a atributos categóricos, sendo que atributos numéricos devem ser reorganizados em classes como *alto*, *médio* e *baixo*, por exemplo. Algumas implementações do algoritmo também continuam a construir a árvore, mesmo que as partições não sejam estatisticamente significantes, posteriormente utiliza-se um algoritmo de poda para reduzir a árvore (BERRY; LINOFF, 2004, tradução nossa).

3.3.5 CART

Classification and Regression Trees (CART) é um algoritmo popular usado na construção de árvores de decisão, publicado em 1984 por Leo Breiman, Jerome Friedman, Richard Olshen e Charles Stone. Utiliza a construção de árvores binárias por

[20] Objeto de saída da execução do algoritmo, também denominado classe (BERRY. LINOFF, 2004, tradução nossa).

meio de partições sobre os atributos, enquanto estas poderem ser feitas. É capaz de detectar sub-árvores que possam representar uma classificação errônea, para que assim seja feito um teste de validação sobre cada nó, identificando-os para em seguida serem retirados da árvore em construção, por meio da poda (BERRY; LINOFF, 2004, tradução nossa).

A Figura 10 explica a ação do algoritmo, que após ser realizada a construção da árvore, verifica-se as possíveis sub-árvores que poderão ser podadas, resultando assim na árvore em destaque.

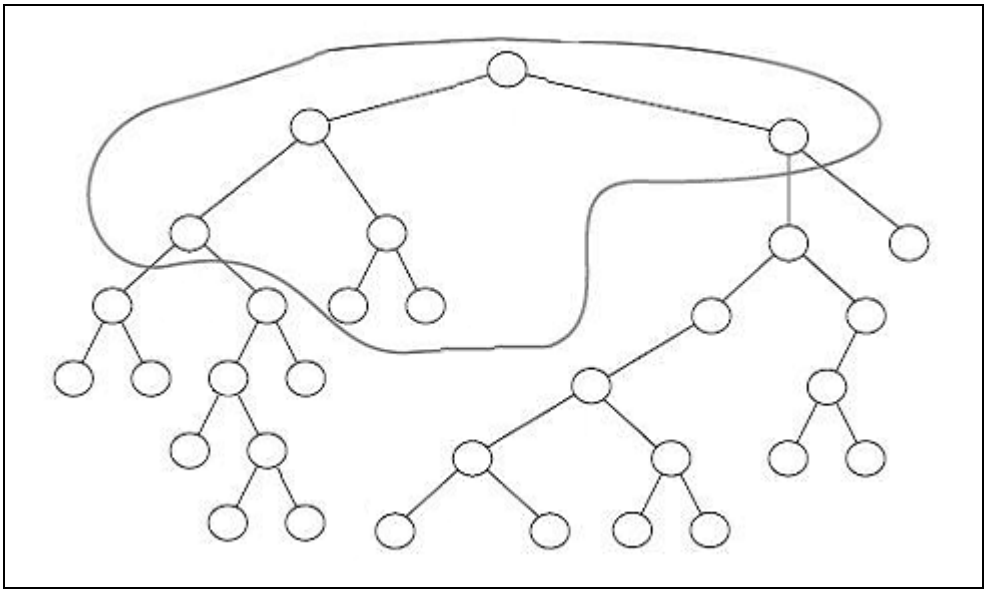

Figura 10. Dentro de uma árvore complexa, existem árvores mais simples e estáveis
Fonte: Adaptado de BERRY, M.; LINOFF, G. (2004)

3.3.6 ID3

Este algoritmo foi desenvolvido e publicado por John Ross Quinlan em meados de 1980. O ID3 constrói uma árvore de decisão baseado num conjunto de tuplas de uma base de dados, onde o atributo classificador de cada tupla seja conhecido. Após a construção da árvore, esta pode ser utilizada para classificar novas tuplas (HAN; KAMBER, 2001, tradução nossa).

A construção da árvore pelo ID3 utiliza uma medida conhecida como ganho de informação (*information gain*), para avaliar qual atributo do conjunto possui maior ganho, construindo um nó na árvore para representar o teste realizado. As ramificações da árvore crescem de acordo com os possíveis valores de cada atributo que é testado, e o conjunto de dados é particionado a partir dessa gama de valores. Em geral, um nó contendo objetos onde todos pertençam a uma mesma classe, é chamado de nó folha. Todo o processo é repetido recursivamente em cada nó não folha, até que não possa mais ser criada nenhuma folha (HAN; KAMBER, 2001, tradução nossa).

3.3.7 C4.5

A publicação deste algoritmo foi realizada no ano de 1987 tendo como desenvolvedor John Ross Quinlan. Este algoritmo é a evolução do ID3, sendo que algumas modificações nas implementações podem ser notadas, como as apresentadas a seguir (QUINLAN, 1993, tradução nossa):

a) mudança na análise do ganho de informação do atributo, denominado de razão do ganho de informação (*gain ratio*). Este modelo possui a mesma equação apresentada no ID3, sendo que ao final utiliza-se uma divisão do ganho de informação gerado pelo atributo em relação a frequência de valores que ocorrem neste;
b) atributos numéricos são particionados em, no máximo, duas ramificações, realizadas na forma A <= Z e A > Z, onde A seria o atributo e Z o valor utilizado para a partição;
c) utilização de valores não informados nos registros da base para realizar o cálculo da razão do ganho de informação, pois juntamente com os

demais valores este torna-se um possível valor a mais encontrado no atributo.

Na Tabela 3 estão relacionadas algumas das características dos algoritmos comentados, a fim de analisar as suas diferenças.

Tabela 3. Resumo dos algoritmos de classificação

ALGORITMO	ANO	ATRIBUTOS	PODA DA ÁRVORE
SLIQ	1996	Nominais e Numéricos	Sim
SPRINT	1996	Nominais e Numéricos	Sim
RAINFOREST	2000	Nominais e Numéricos	Sim
CHAID	1975	Nominais e Numéricos	Não
CART	1984	Numéricos	Sim
ID3	anos 80	Nominais	Não
C4.5	1987	Nominais e Numéricos	Sim

Dentre os algoritmos apresentados, utilizou-se nesta pesquisa o C4.5. Assim, as suas características e funcionalidades encontram-se apresentados no próximo capítulo.

4 O ALGORITMO C4.5

O algoritmo tem como objetivo gerar um modelo classificador na forma de uma árvore de decisão, apresentando dois estados durante o processo que são (QUINLAN, 1993, tradução nossa):

a) folha que indica um ponto final da classificação, sendo atribuída a uma classe;
b) nó de decisão, onde baseando-se no atributo em análise, poderá conter uma ramificação seguida de uma folha ou uma sub-árvore para cada possível valor encontrado na base.

Inicialmente o algoritmo necessita ter alguns dos seus parâmetros informados para execução, sendo indispensável a indicação do conjunto de dados que será utilizado. Após ser informada a fonte de informações, o algoritmo é executado em duas etapas:

a) **construção da árvore de decisão:** é realizada a separação dos dados em duas ou mais partições a fim de gerar uma árvore, utilizando algumas restrições sobre os conjuntos de valores de cada atributo. O processo deve ser repetido recursivamente até que todos ou a maioria dos exemplos dos dados de cada partição sejam participantes de uma classe em particular. A árvore armazena todo o conjunto dos dados, sendo construída em largura, a fim de que todos os nós de um determinado nível da árvore sejam processados antes de iniciar o processo do nível subsequente (GOLDSCHMIDT; PASSOS, 2005);
b) **simplificação da árvore de decisão:** por meio de medidas estatísticas avalia-se a significância de algumas regras geradas pela árvore. Aquelas

> que não acrescentam conhecimento são podadas, resultando em uma árvore rápida e de correta classificação (HAN; KAMBER, 2001, tradução nossa).

Utilizando o conceito de Quinlan (1993, tradução nossa) juntamente com o de Goldschmidt e Passos (2005) sobre o algoritmo C4.5, que explica sobre as etapas e cálculos realizados de uma forma mais didática, demonstra-se a seguir o funcionamento do algoritmo.

Considere a realização da tarefa de classificação em um conjunto $S(A_1, A_2, ..., A_n, C)$, onde C é o atributo rotulador, que possui os valores: $C_1, C_2, ..., C_k$, que são as classes do sistema (QUINLAN, 1993, tradução nossa).

Durante o processo de construção da árvore deve-se seguir duas operações (GOLDSCHMIDT; PASSOS, 2005):

a) avaliação dos pontos de separação de cada nó interno da árvore e a identificação de qual o melhor ponto de separação;
b) criação das partições usando o melhor ponto de separação identificado para os casos pertencentes a cada nó. Após a identificação do melhor ponto de separação de cada nó, aplica-se o critério identificado para a criação das partições.

A fim de avaliar os pontos de separação de cada nó interno da árvore, calculam-se as seguintes medidas (QUINLAN, 1993, tradução nossa):

a) ganho de informação considerando a partição da base de dados associada ao nó em análise. Observa-se que para o nó raiz, a base de dados está completa. Sendo assim, deve-se utilizar a seguinte equação sobre o atributo de classificação:

$$info(S) = -\sum_{j=1}^{k} \frac{freq(C_j, S)}{|S|} \times log_2\left(\frac{freq(C_j, S)}{|S|}\right) bits \quad (1)$$

Onde:

- *S* representa a partição da base de dados,

- *freq(C_j ,S)* representa o número de vezes que a classe C_j acontece em *S*,

- |*S*| denota o número de casos do conjunto *S*,

- *k* indica o número de classes distintas,

- *bits* unidade de medida utilizada ao referir-se a valores de informação;

b) ganho de informação de cada atributo considerando a partição da base de dados associada ao nó em análise. Observa-se que, para o nó raiz, todos os atributos, com exceção do atributo de classificação, devem ser analisados com as seguintes equações:

$$info_x(T) = \sum_{i=1}^{n} \frac{|T_i|}{|T|} \times info(T_i) \quad (2)$$

Onde:

- *T* refere-se a quantidade de ocorrências na partição em análise,

- T_i representa a quantidade de ocorrências de uma classe contida no conjunto *T*;

$$gain(T) = info(T) - info_x(T) \quad (3)$$

c) razão do ganho de informação por meio dos valores obtidos pelo ganho de informação das classes e do atributo, analisando-se por meio das seguintes equações:

$$split\ info(X) = -\sum_{i=1}^{n} \frac{|T_i|}{|T|} \times \log_2\left(\frac{|T_i|}{|T|}\right) \quad (4)$$

$$gain\ ratio(X) = gain(X) / split\ info(X) \quad (5)$$

d) seleção do atributo com maior razão de ganho de informação obtido sobre a partição em análise.

A fim de compreender o processo de atribuição dos pontos de separação, necessita-se destacar que esta etapa depende do domínio de cada atributo, podendo ele ser numérico ou categórico. No caso do atributo ser numérico, a partição será realizada em duas ramificações, sendo o nó identificado como menor ou igual ao ponto de separação escolhido, ou maior que este valor. Já por outro lado, caso o atributo seja categórico, a partição irá conter uma ramificação para cada valor do atributo (QUINLAN, 1993, tradução nossa).

As etapas realizadas pelo algoritmo C4.5 na construção da árvore estão demonstradas na Figura 11 em forma de pseudo-código dos comandos executados.

```
1)  Gerar_arvore_decisao(si, lista_de_atributos)
2)    criar um nó N;
3)    se registros são todos da mesma classe C então
4)      retorna N como uma folha e atribua a classe C como identificador
5)    se registros estão vazios então
6)      retorna N como uma folha e atribui a classe mais comum no nó anterior como identificador
7)    seleciona o atributo com maior ganho de informação para particionamento do nó
8)    identifique o nó N pelo atributo com maior ganho de informação
9)    para cada valor conhecido ai do atributo
10)     construa uma ramificação no nó N para cada condição atributo = ai
11)     grave si com os registros até então classificados para o atributo = ai
12)     se si estiver vazio então
13)       grave uma folha ao nó identificado com a classe mais comum dos registros
14)     senão grave um nó retornado por Gerar_arvore_decisao(si, lista_de_atributos - atributo)
```

Figura 11. Pseudo-código do algoritmo para indução de árvores de decisão
Fonte: Adaptado de HAN, J.; KAMBER, M. (2001, tradução nossa)

Utilizando os dados da Tabela 4, ilustra-se o procedimento descrito anteriormente.

Tabela 4. Conjunto de dados para avaliar se a condição climática está favorável à prática de golfe

APARÊNCIA	TEMP (℉)	UMIDADE (%)	VENTO	JOGAR GOLFE
Ensolarada	75	70	Sim	Sim
Ensolarada	80	90	Sim	Não
Ensolarada	85	85	Não	Não
Ensolarada	72	95	Não	Não
Ensolarada	69	70	Não	Sim
Nublada	72	90	Sim	Sim
Nublada	83	78	Não	Sim
Nublada	64	65	Sim	Sim
Nublada	81	75	Não	Sim
Chuvosa	71	80	Sim	Não
Chuvosa	65	70	Sim	Não
Chuvosa	75	80	Não	Sim
Chuvosa	68	80	Não	Sim
Chuvosa	70	96	Não	Sim

Fonte: GOLDSCHMIDT, R.; PASSOS, E.(2005).

Considere *S* o conjunto de dados que contém informações sobre as condições climáticas e a recomendação a jogar ou não golfe. Em cada situação, sendo apresentado o atributo *temperatura* interpretado pela abreviação *TEMP (ºF)*, objetiva-se construir uma árvore de decisão que ofereça a recomendação quanto à prática de golfe.

No exemplo, o atributo de classificação é *Jogar Golfe*, já que esta é a saída que se deseja obter por meio da construção da árvore. Ele contém duas classes, *Sim* com nove ocorrências e *Não* com cinco ocorrências. Calcula-se o ganho de informação do conjunto *S* completo, por meio da equação:

- conjunto de dados = 14 registros,

- frequência da classe *jogar_golfe* (sim) = 9 registros,

- frequência da classe *jogar_golfe* (não) = 5 registros.

$$info(S) = -\sum_{j=1}^{k} \frac{freq(C_j, S)}{|S|} \times log_2\left(\frac{freq(C_j, S)}{|S|}\right) bits$$

$$info(jogar\_golfe) = -9/14 \times \log_2(9/14) - 5/14 \times \log_2(5/14) = 0{,}94\, bits$$

Na sequência, considerando o atributo *Aparência*, deve-se dividir o conjunto *S* em três partições, *T1*, *T2*, *T3*, representando os valores possíveis do atributo (*Ensolarada*, *Nublada* e *Chuvosa*). O cálculo do ganho de informação deste atributo é realizado utilizando-se a Equação (2) e (3):

$$info_x(T) = \sum_{i=1}^{n} \frac{|T_i|}{|T|} \times info(T_i)$$

info(aparência) = 5/14 x (–2/5 x log2 (2/5) – 3/5 x log2 (3/5)) +

4/14 x (–4/4 x log2 (4/4) – 0/4 x log2 (0/4)) +

5/14 x (–3/5 x log2 (3/5) – 2/5 x log2 (2/5)) = 0,694 *bits*

$$gain(T) = info(T) - info_x(T)$$

gain(aparência) = 0,940 – 0,694 = 0,246*bits*

Após ter sido encontrado o ganho total para o atributo *Aparência*, por meio da Equação (4) e (5) realiza-se o cálculo da razão do ganho de informação:

$$split\ info(X) = -\sum_{i=1}^{n} \frac{|T_i|}{|T|} \times log_2\left(\frac{|T_i|}{|T|}\right)$$

$$\begin{aligned} split\ info(X) = &(-5/14) \times \log_2(5/14) + \\ &(-4/14) \times \log_2(4/14) + \\ &(-5/14) \times \log_2(5/14) = 1{,}577 bits \end{aligned}$$

$$gain\ ratio(X) = gain(X) / split\ info(X)$$

$$gain\ ratio(X) = 0{,}246 / 1{,}577 = 0{,}156 bits$$

O cálculo da razão do ganho de informação deve ser repetido de maneira semelhante para os demais atributos presentes no conjunto de dados. O atributo com maior razão do ganho ao final desta etapa será o ponto de separação do nó raiz da árvore. Neste momento a base de dados é particionada e repete-se o processo para cada novo nó gerado.

Quinlan (1993, tradução nossa), o autor do algoritmo, salienta sobre a utilização do cálculo para a partição pelo ganho de informação ou pela razão do ganho de informação. Estudos realizados com o auxílio do C4.5 obtiveram resultados variados no uso dos métodos citados. Algumas bases de dados com um número menor de registros a serem classificados acabam gerando informações incorretas e árvores de decisão reduzidas quando utilizado o método da razão do ganho de informação. Assim, disponibiliza-se ao usuário caso ele deseje que o algoritmo utilize um ou outro método para sua execução.

Ao ser encerrada a construção inicial do modelo classificador o algoritmo C4.5 o analisa a fim de avaliar sua confiabilidade, utilizando-se do cálculo denominado de poda pessimista. Na poda pessimista calcula-se o limite superior entre a taxa de erro e a quantidade de casos de um determinado nó multiplicado pelo nível de confiança desejado pelo usuário. Efetuando-se testes entre os valores encontrados e aqueles obtidos na taxa de erro de cada nó, caso esta for superior ao coeficiente calculado para a poda, o nó será então podado (QUINLAN, 1993, tradução nossa).

A equação é descrita como $U_{CF}(E, N)$, onde:

- E taxa de erro encontrada na folha;

- N quantidade de casos encontrados na folha.

Esta equação é utilizada para verificar se os nós da árvore possuem folhas que obedecem ao nível de confiança estipulado pelo usuário ou não, sendo podadas as folhas que não satisfaçam o cálculo. Assim, transforma-se o nó anteriormente testado em uma folha da árvore. No exemplo utilizado com a base de dados golfe, a etapa da poda não resulta em modificação aplicada ao modelo classificador, pois as taxas de erro encontradas satisfazem o nível de confiança padrão de 25%.

A Figura 12 apresenta o pseudo-código dos processos executados na etapa de realização da poda da árvore, após ter sido concluído o processo de sua construção.

```
1)  poda(nó)
2)    se nó = INTERIOR então
3)      poda(nó.filho_esquerdo)
4)      poda(nó.filho_direito)
5)      se subárvore_erro(nó) > erro(nó) então
6)        grave tipo do nó como FOLHA
7)
8)  subárvore_erro(nó)
9)     e = nó.filho_esquerdo
10)   d = nó.filho_direito
11)   se nó = INTERIOR então
12)     retorna (e.registros * subárvore_erro(e)) +
13)                (d.registros * subárvore_erro(d))
14)   senão
15)     retorna erro(nó)
```

Figura 12. Pseudo-código do algoritmo de realização de poda da árvore
Fonte: Adaptado de WITTEN, I. H.; FRANK, E. (2005, tradução nossa)

A seguir demonstra-se a partição dos dados conforme cada regra gerada pelo modelo classificador criado a partir da base de dados golfe.

4.1 PARTIÇÃO DO CONJUNTO DE DADOS

A árvore gerada pela execução do algoritmo C4.5 irá resultar em três partições de primeiro nível e duas de segundo. Tendo-se a seguinte estrutura da árvore:

a) **nó aparência e umidade:** caso a aparência seja *ensolarada*, então deverá ser verificado se a umidade relativa do ar está igual ou abaixo dos 75%, sendo neste caso possível a realização de uma partida de golfe. Caso contrário, se estiver acima dos 75%, não deverá ocorrer a partida. Os registros da base de dados golfe pertencentes a estas regras podem ser visualizados nas Tabelas 5 e 6;

- Aparência = Ensolarada

  Umidade <= 75: Jogar Golfe = Sim

Tabela 5. Dados classificados na primeira folha da árvore

APARÊNCIA	TEMP (℉)	UMIDADE (%)	VENTO	JOGAR
Ensolarada	75	70	Sim	Sim
Ensolarada	69	70	Não	Sim

Fonte: GOLDSCHMIDT, R.; PASSOS, E.(2005).

Umidade > 75: Jogar Golfe = Não

Tabela 6. Dados classificados na segunda folha da árvore

APARÊNCIA	TEMP (℉)	UMIDADE (%)	VENTO	JOGAR
Ensolarada	80	90	Sim	Não
Ensolarada	85	85	Não	Não
Ensolarada	72	95	Não	Não

Fonte: GOLDSCHMIDT, R.; PASSOS, E.(2005).

b) **nó aparência:** caso esteja *nublada* poderá ser realizada a partida de golfe. Os dados neste nó podem ser visualizados na Tabela 7;

- Aparência = Nublada: Jogar Golfe = Sim

Tabela 7. Dados classificados na terceira folha da árvore

APARÊNCIA	TEMP (℉)	UMIDADE (%)	VENTO	JOGAR
Nublada	72	90	Sim	Sim
Nublada	83	78	Não	Sim
Nublada	64	65	Sim	Sim
Nublada	81	75	Não	Sim

Fonte: GOLDSCHMIDT, R.; PASSOS, E.(2005).

c) **nó aparência e vento:** caso o atributo vento seja *sim*, neste caso será melhor deixar a partida de golfe para outro momento. Caso contrário, se o atributo vento for *não*, pode-se realizar a partida (Tabelas 8 e 9).

- Aparência = Chuvosa

  Vento = Sim: Jogar Golfe = Não

Tabela 8. Dados classificados na quarta folha da árvore

APARÊNCIA	TEMP (℉)	UMIDADE (%)	VENTO	JOGAR
Chuvosa	71	80	Sim	Não
Chuvosa	65	70	Sim	Não

Fonte: GOLDSCHMIDT, R.; PASSOS, E.(2005).

Vento = Não: Jogar Golfe = Sim

Tabela 9. Dados classificados na quinta folha da árvore

APARÊNCIA	TEMP (℉)	UMIDADE (%)	VENTO	JOGAR
Chuvosa	75	80	Não	Sim
Chuvosa	68	80	Não	Sim
Chuvosa	70	96	Não	Sim

Fonte: GOLDSCHMIDT, R.; PASSOS, E.(2005).

Demonstrada a partição dos dados e as devidas ramificações encontradas na árvore construída, tem-se a seguir o modelo completo construído.

4.2 ÁRVORE DE DECISÃO CONSTRUÍDA

A realização da construção do modelo classificador, a partir do exemplo contido na Tabela 4, resultou na árvore visível na Figura 13.

```
Aparência = Ensolarada:
        Umidade <= 75: Jogar Golfe = Sim
        Umidade > 75: Jogar Golfe = Não
Aparência = Nublada: Jogar = Sim
Aparência = Chuvosa;
        Vento = Sim: Jogar Golfe = Não
        Vento = Não; Jogar Golfe = Sim
```

Figura 13. Árvore textual construída a partir da base de dados golfe
Fonte: Adaptado de QUINLAN, J. R. (1993, tradução nossa)

Na Figura 14 está montada graficamente a árvore resultante do modelo classificador obtido pelos dados da Tabela 4.

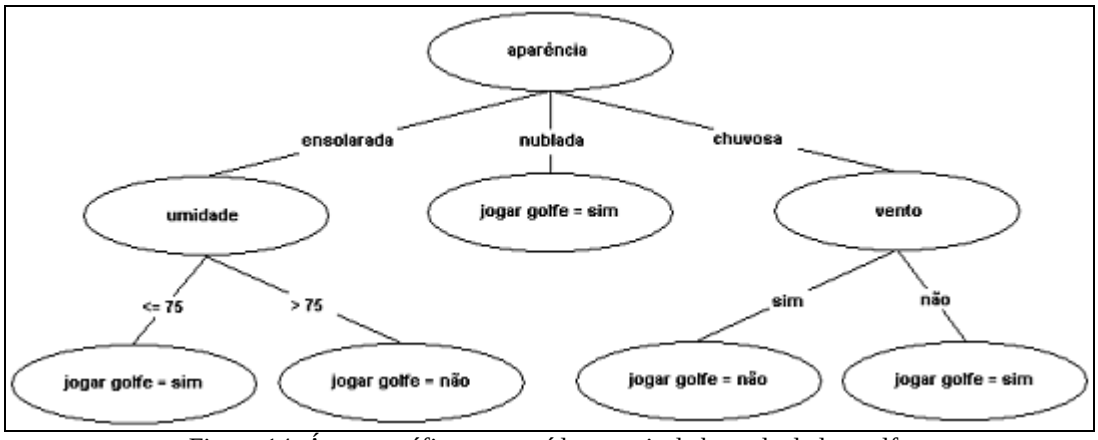

Figura 14. Árvore gráfica construída a partir da base de dados golfe

A tarefa de classificação pelo método de indução de árvores por meio do algoritmo C4.5 é utilizada em ferramentas disponíveis na internet, tanto gratuitas como comerciais. A demonstração do funcionamento destas ferramentas e como estão estruturadas para a utilização pelo usuário, seus processos e parâmetros, devem ser analisados a fim de ser implementado o módulo do algoritmo na *Shell* Orion.

5 ALGUNS EXEMPLOS DE USO DO ALGORITMO DE CLASSIFICAÇÃO C4.5

Aplicações utilizando *data mining* são encontradas em diversas áreas de conhecimento, desde campanhas de marketing a investigações criminais, auxiliando tomadas de decisões com a descoberta de relações entre os dados (KANTARDZIC, 2003, tradução nossa).

Para o desenvolvimento do trabalho proposto e visando conhecer o funcionamento de ferramentas que utilizam *data mining* com o algoritmo C4.5, a seguir encontram-se alguns exemplos analisados.

5.1 WEKA

Desenvolvida pela Universidade de Waikato localizada na Nova Zelândia, a ferramenta Weka[21], que é distribuída gratuitamente, possui os mais diferentes métodos e tarefas implementadas para a busca de conhecimento em variadas bases de dados (WAIKATO, 2005, tradução nossa).

A ferramenta possibilita a busca dos dados para mineração por meio de arquivos ou bancos de dados, onde para estes existe uma tela de configuração das conexões, que possibilita ao usuário utilizar distribuições de diferentes fabricantes (WAIKATO, 2005, tradução nossa).

Com a opção de pré-processamento, a ferramenta também disponibiliza a capacidade do usuário utilizar uma base de dados filtrada de informações que possam

[21] Disponível em http://www.cs.waikato.ac.nz/ml/weka

conter inconsistências ou apresentar ausência de valores (WAIKATO, 2005, tradução nossa).

A ferramenta Weka disponibiliza para uso entre os algoritmos um classificador chamado J48, que é a denominação para a versão 8 do algoritmo C4.5 implementado na linguagem JAVA (WAIKATO, 2005, tradução nossa).

Na Figura 15 se apresenta o ambiente de exploração das tarefas existentes nesta ferramenta.

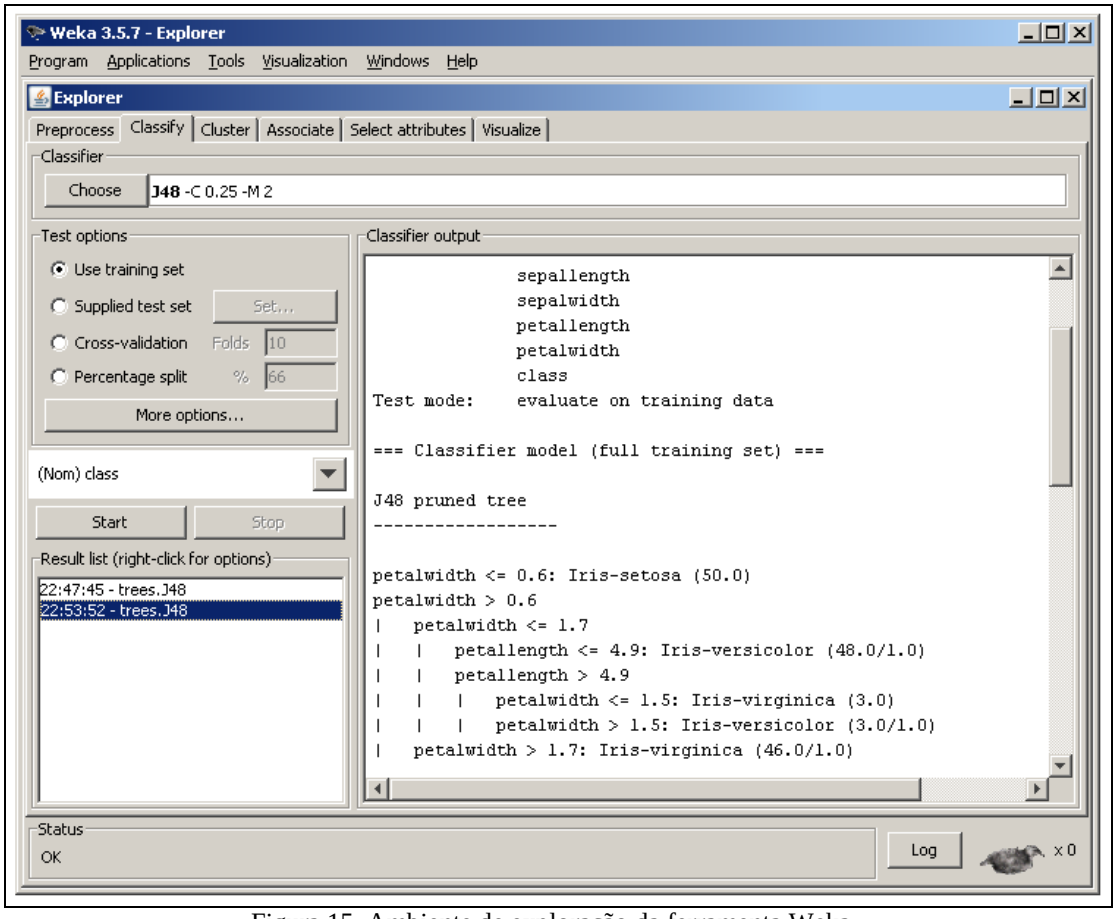

Figura 15. Ambiente de exploração da ferramenta Weka
Fonte: Adaptado de WAIKATO (2005, tradução nossa)

5.2 ODBCMINE

A ferramenta ODBCMINE[22] foi desenvolvida pela empresa Intelligent Systems Research LLC e possui o algoritmo C4.5 como método de classificação dos dados. As conexões a bases de dados podem ser realizadas no MS Access e Microsoft Text ODBC Driver, que são as duas ferramentas até então testadas pela empresa. A distribuição não é gratuita, mas disponibiliza uma versão de testes capaz de trabalhar com quatro atributos e quinze registros no máximo (INTELLIGENT SYSTEMS RESEARCH LLC, 2009).

A ferramenta é utilizada apenas com parâmetros via comando para realizar a execução da tarefa, sendo que não há uma parte gráfica para apresentar seus resultados. O diferencial da ferramenta está na amostragem dos resultados obtidos, que é realizada no formato de árvores de decisões gráficas em *Scalable Vector Graphics*[23] (SVG) (INTELLIGENT SYSTEMS RESEARCH LLC, 2009).

Na Figura 16 tem-se a visualização dos resultados gerados pela ferramenta utilizando o Internet Explorer 6.0.

[22] Disponível em http://www.intsysr.com

[23] Modelo de apresentação de árvores de decisão podendo ser visualizado graficamente utilizando o browser da internet (INTELLIGENT SYSTEMS RESEARCH LLC, 2009). Disponível para *download* em http://www.adobe.com

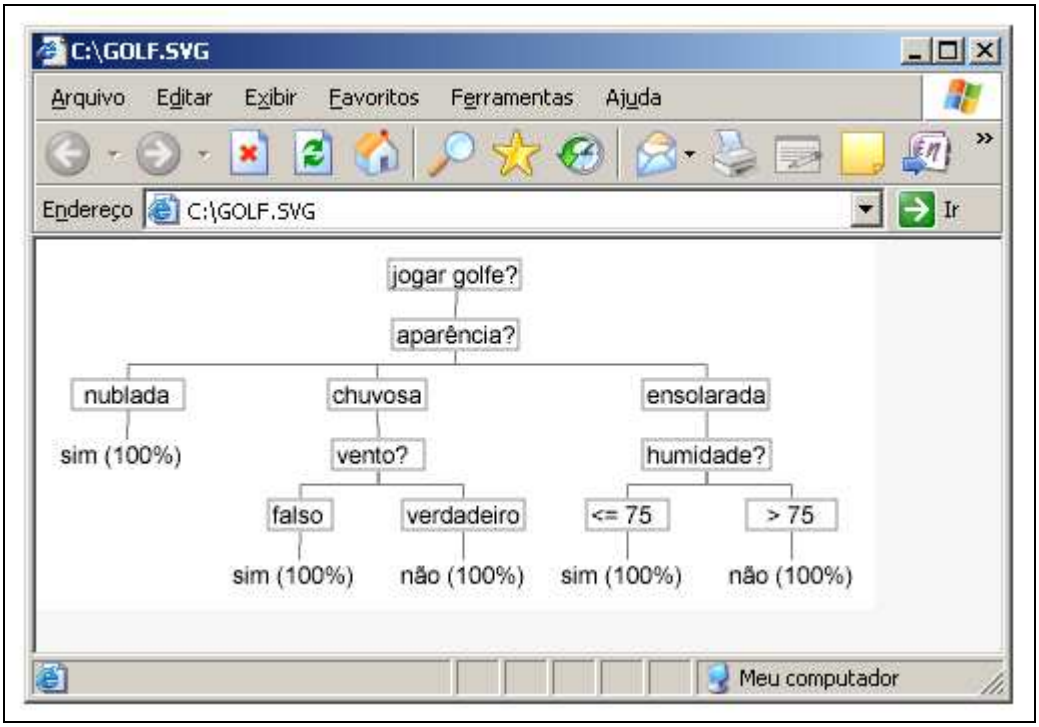

Figura 16. Resultados gerados na ferramenta ODBCMINE pela base de dados golfe
Fonte: Traduzido de INTELLIGENT SYSTEMS RESEARCH LLC (2009)

5.3 C4.5

O autor do algoritmo C4.5[24], Quinlan, disponibiliza gratuitamente o código-fonte para download. Por meio deste e de uma ferramenta de desenvolvimento na linguagem C, o código é possível ser compilado a fim de gerar o executável e realizar testes com algumas bases de dados distribuídas junto ao fonte (QUINLAN, 1993, tradução nossa).

A ferramenta utiliza na busca de dados, atributos e classes para a classificação arquivos formatados com a extensão .data e .names, no qual o primeiro possui os dados seqüencialmente distribuídos na ordem dos atributos e classes especificadas no segundo arquivo (QUINLAN, 1993, tradução nossa).

[24] Disponível em http://www.rulequest.com/Personal/

Nas Figuras 17 e 18 é possível identificar como estão apresentadas as informações contidas nos arquivos *.data* e *.names*.

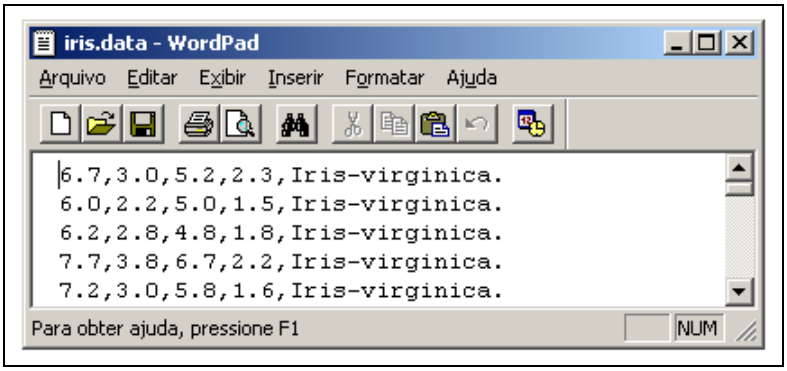

Figura 17. Informações contidas em um arquivo *.data*
Fonte: Adaptado de QUINLAN, J. R. (1993, tradução nossa)

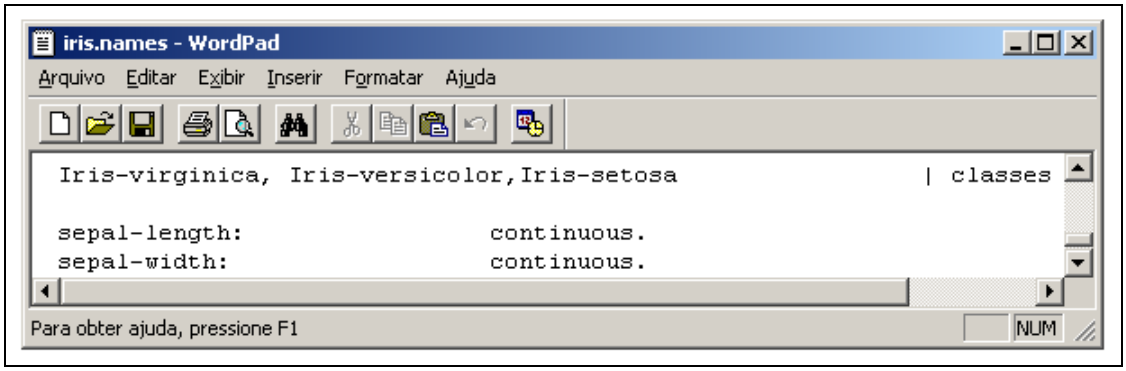

Figura 18. Informações contidas em um arquivo *.names*
Fonte: Adaptado de QUINLAN, J. R. (1993, tradução nossa)

A execução da ferramenta é realizada no ambiente texto do sistema operacional, onde, no exemplo visualizado na Figura 19, encontra-se os resultados obtidos ao ser utilizada a base de dados iris.

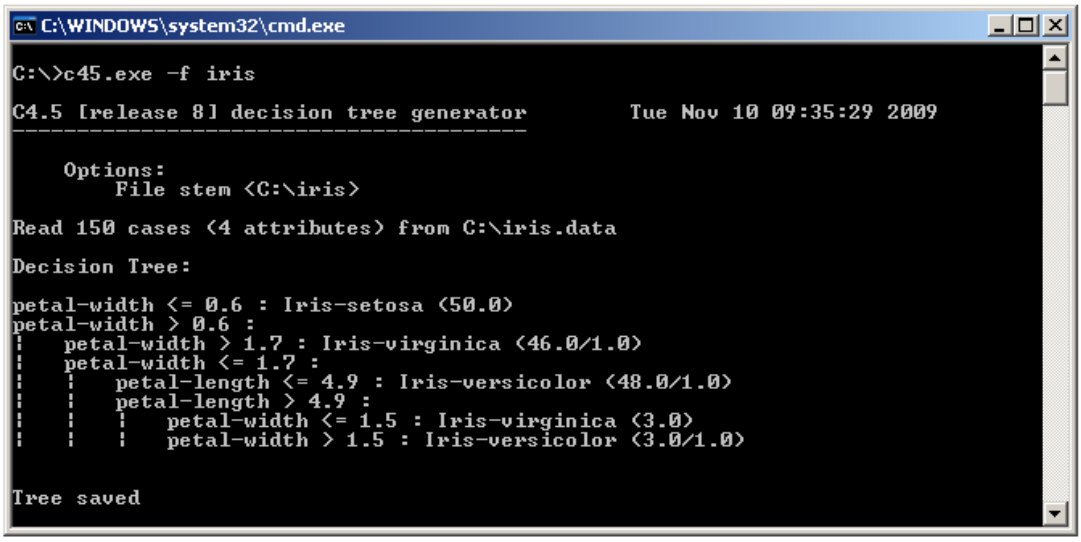

Figura 19. Resultado da ferramenta C4.5 utilizando a base de dados Iris
Fonte: Adaptado de QUINLAN, J. R. (1993)

A realização do estudo das ferramentas torna-se indispensável para a implementação do módulo de classificação pelo algoritmo C4.5 na *Shell* Orion. Porém, antes da sua implementação, necessita-se conhecer a estrutura da S*hell* Orion e os módulos que já encontram-se disponíveis.

6 A *SHELL* DE *DATA MINING* ORION

A *Shell* Orion encontra-se em desenvolvimento pelo Grupo de Pesquisa em Inteligência Computacional Aplicada do curso de Ciência da Computação da UNESC. Na Tabela 10 tem-se os módulos desenvolvidos e os acadêmicos que participaram da construção desta ferramenta.

Tabela 10. Algoritmos implementados na *Shell* Orion

Acadêmico	Algoritmo	Módulo	Ano
Diego Paz Casagrande	*Apriori*[25]	Associação	2005
Diana Colombo Pelegrin	ID3	Classificação	2005
Leandro Sehnem Bortolotto	*Kohonen*[26]	Clusterização	2007
Dênis Piazza Martins	*K-means*	Clusterização	2007
Lidiane Rosso Raimundo	CART	Classificação	2007
José Márcio Cassettari Júnior	*Gustafson-Kessel*[27]	Clusterização	2008
Daniel Perego	*Gath-Geva*[28]	Clusterização	2009

Os trabalhos inicialmente desenvolvidos na *shell* implementaram a capacidade de escolha de conexão com os bancos de dados: PostgreSQL[29] e FireBird[30]. Percebendo a necessidade em adaptar a ferramenta a conexão com diferentes bases de dados, alunos e orientadores resolveram padronizá-la quanto a esta questão. Como é possível verificar na Figura 20, as conexões podem ser cadastradas na *shell*, informando um nome, o *driver* e a url, onde a base de dados se encontra.

[25] Algoritmo de associação que procura por conjuntos freqüentes de dados de maneira recursiva (CASAGRANDE, 2005).

[26] Utiliza o método de redes neurais artificiais a fim de obter resultados de agrupamento entre os dados para a tarefa de clusteriação (BORTOLOTTO, 2007).

[27] Por meio do método de lógica *fuzzy*, este algoritmo tem como objetivo agrupar dados com características semelhantes, sendo também capaz de permitir um elemento pertencer a mais de um grupo simultaneamente (CASSETTARI JÚNIOR, 2008).

[28] Evolução do algoritmo *Gustafson-Kessel*, que tem por diferença a capacidade de obter *clusters* com menor erro de aproximação entre os dados (PEREGO, 2009).

[29] Disponível gratuitamente em http://www.postgresql.org

[30] Disponível gratuitamente em http://www.firebirdsql.org

Figura 20. Cadastro de conexões da *Shell* Orion
Fonte: BORTOLOTTO, L. S. (2007)

Na Figura 21 visualiza-se a tela de configurações de conexão da base de dados, onde o usuário primeiramente indica qual a conexão, já cadastrada, ele irá utilizar, informando após isto também o nome da base, usuário e senha para conectar-se.

Figura 21. Parâmetros de conexão a base de dados
Fonte: BORTOLOTTO, L. S. (2007)

Após ser realizada a etapa de conexão à base de dados o usuário estará apto a utilizar qualquer um dos métodos e algoritmos disponibilizados na ferramenta.

Na Figura 22 e 23 tem-se os módulos dos algoritmos *Apriori* e ID3 simultaneamente, sendo esses os primeiros implementados na *Shell* Orion.

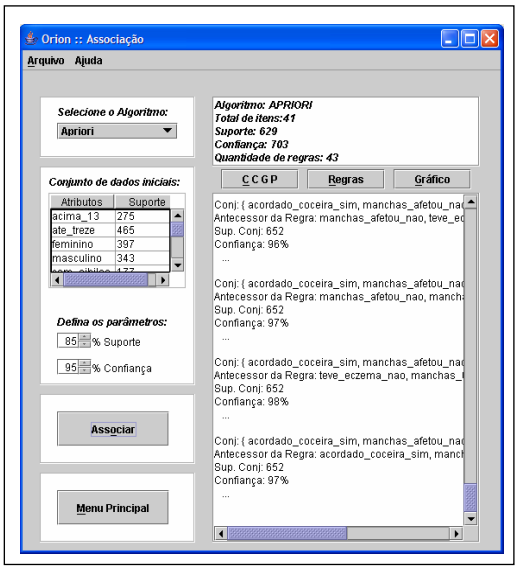

Figura 22. Associação pelo algoritmo *Apriori*
Fonte: CASAGRANDE, D. P. (2005)

Figura 23. Classificação pelo algoritmo ID3
Fonte: PELEGRIN, D. C. (2005)

O algoritmo CART apresenta os parâmetros e os resultados na mesma tela, assim como o ID3, visível na Figura 24. Os resultados deste algoritmo podem ser visualizados em modo texto como um relatório geral do seu processamento, em forma de regras e também como uma árvore de decisão.

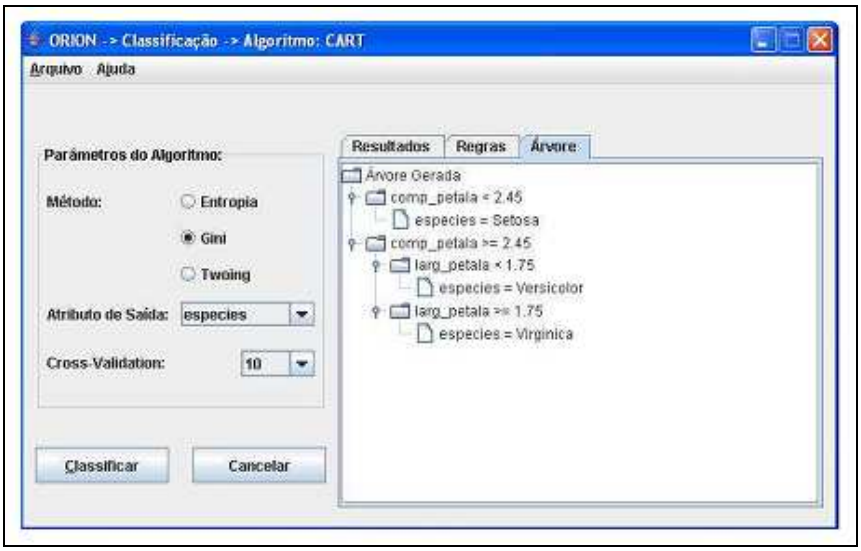

Figura 24. Algoritmo CART na *Shell* Orion
Fonte: RAIMUNDO, L. R. (2007)

O módulo de clusterização possui implementado os algoritmos *Kohonen*, *K-means*, *Gustafson-Kessel* e o *Gath-Geva*. A Figura 25 demonstra a parametrização

utilizada no algoritmo *Kohonen*, sendo que na Figura 26 apresentam-se os resultados por meio de gráfico, árvore de grupos, comandos SQL e um relatório geral do processo em modo texto.

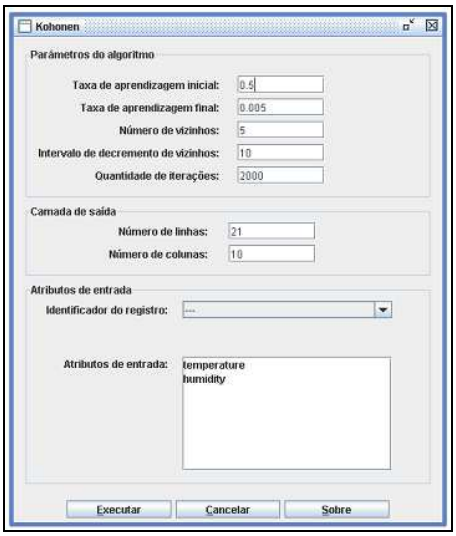

Figura 25. Parâmetros do algoritmo *Kohonen*
Fonte: BORTOLOTTO, L. S. (2007)

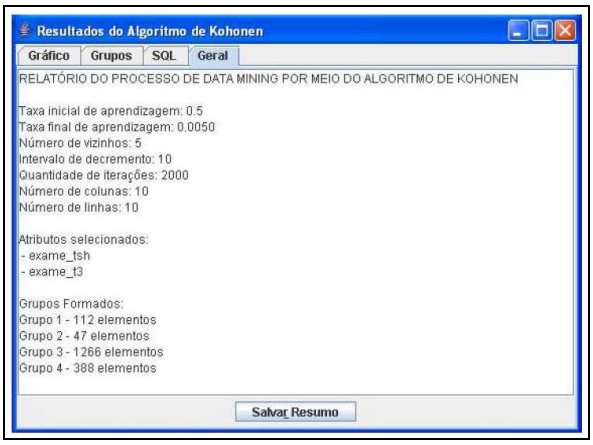

Figura 26. Resultados do algoritmo *Kohonen*
Fonte: BORTOLOTTO, L. S. (2007)

A tarefa de clusterização por meio do algoritmo *K-means* está demonstrada na Figura 27, tendo-se a parametrização para a sua execução na mesma tela de visualização dos resultados obtidos.

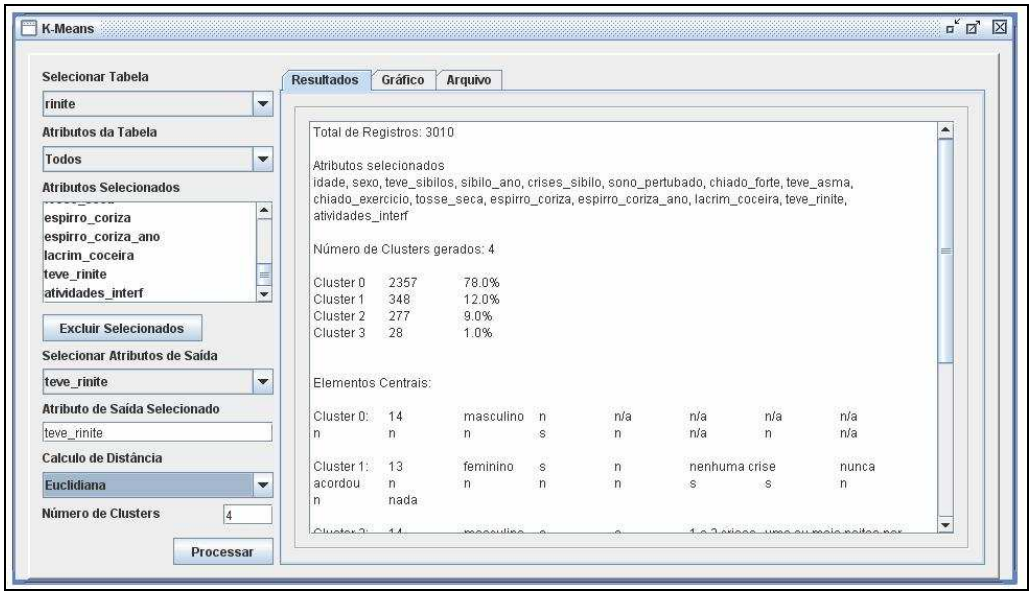

Figura 27. Parâmetros e resultados do algoritmo *K-Means*
Fonte: MARTINS, D. P. (2007)

O algoritmo *Gustafson-Kessel* possui uma tela inicial com seus parâmetros (Figura 28). A execução deste disponibiliza os resultados em outra tela, como pode ser visto na Figura 29, em forma de gráfico, árvore de grupos, comandos SQL e relatório em modo texto sobre o processamento realizado.

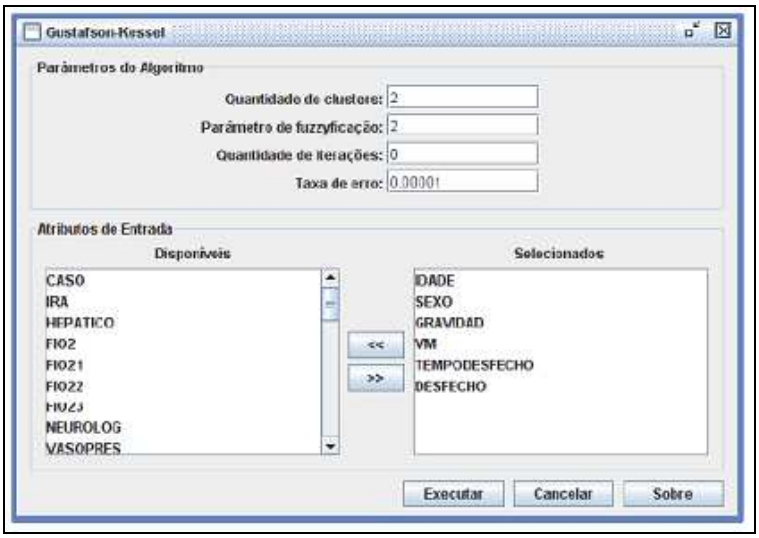

Figura 28. Parâmetros do algoritmo *Gustafson-Kessel*
Fonte: CASSETTARI JÚNIOR, J. M. (2008)

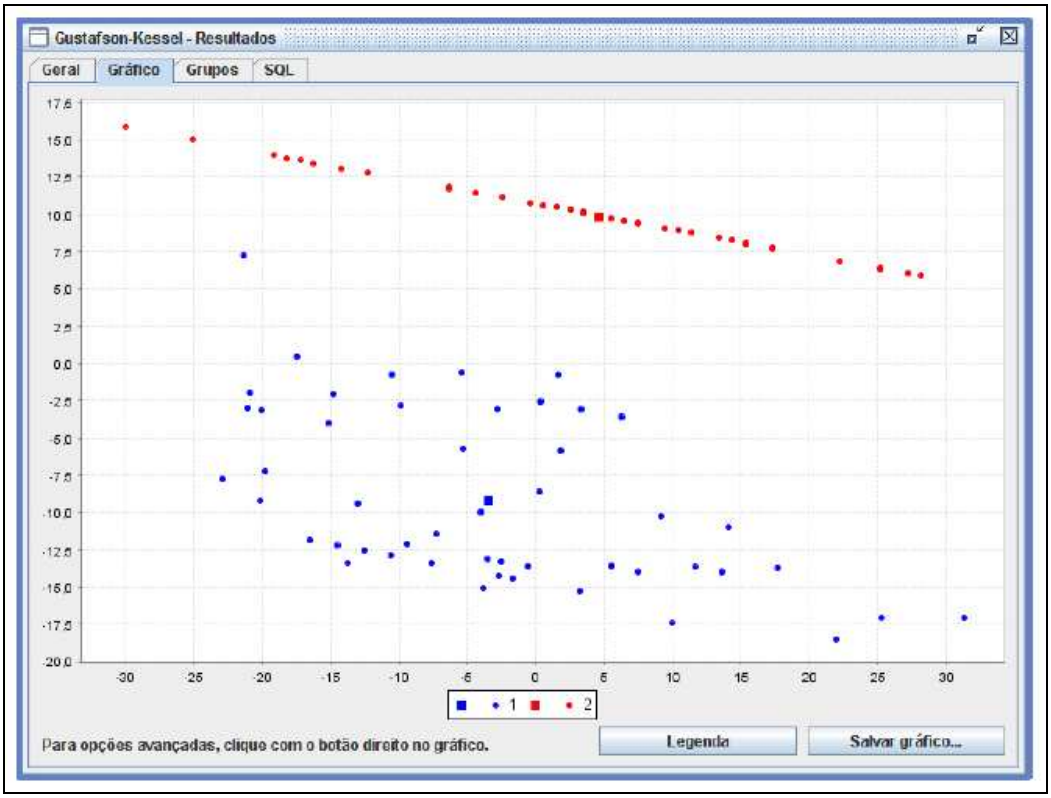

Figura 29. Resultados apresentados na execução do algoritmo *Gustafson-Kessel*
Fonte: CASSETTARI JÚNIOR, J. M. (2008)

O algoritmo *Gath-Geva*, que também realiza a tarefa de clusterização, possui as funcionalidades parecidas com as do algoritmo *Gustafson-Kessel*, sendo as telas de parâmetros e resultados praticamente idênticas. Na Figura 30 visualiza-se os resultados em forma de gráfico pelo algoritmo *Gath-Geva*.

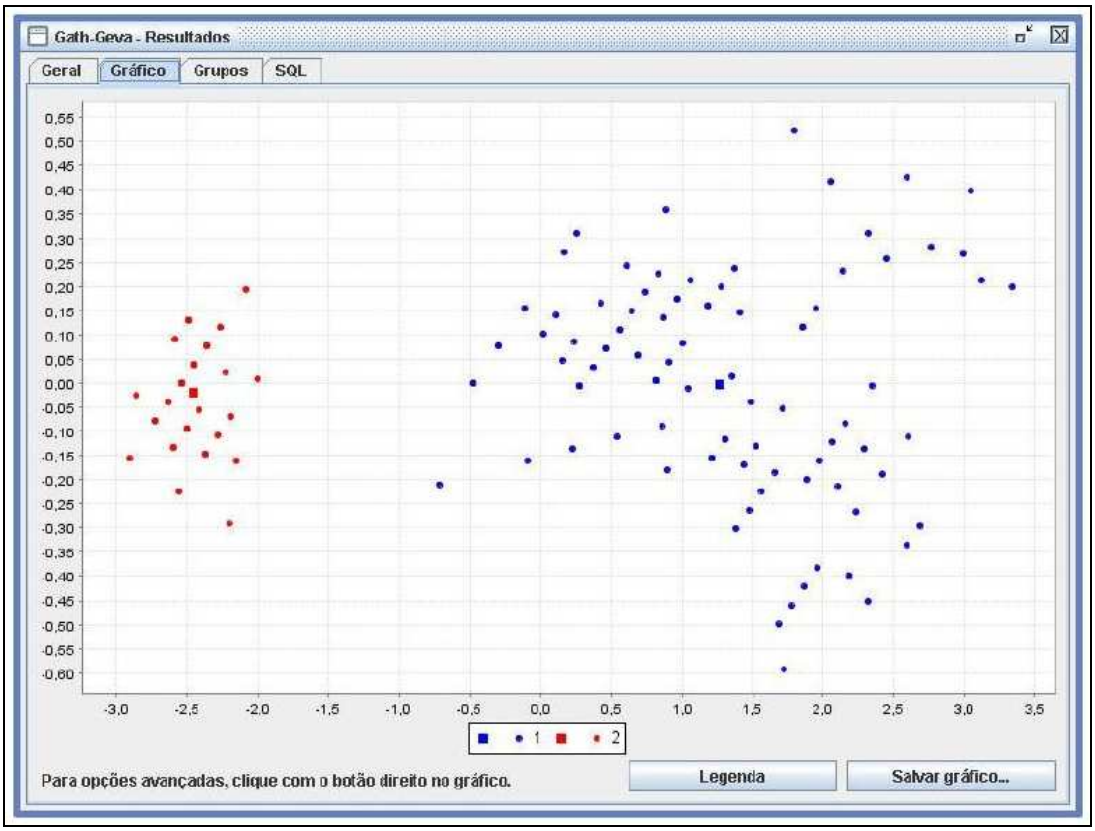

Figura 30. Resultados apresentados na execução do algoritmo *Gath-Geva*
Fonte: PEREGO, D. (2009)

Utilizando os conceitos de *data mining*, os módulos presentes realizam suas funções a fim de descobrir relações e/ou conhecimento nas bases de dados testadas. Esta pesquisa compreendeu a inserção do algoritmo C4.5 no módulo de classificação da *Shell* Orion, tendo-se a seguir a descrição do trabalho desenvolvido.

7 O ALGORITMO C4.5 NA TAREFA DE CLASSIFICAÇÃO DA *SHELL* ORION *DATA MINING ENGINE*

As pesquisas realizadas pelos acadêmicos e professores para o desenvolvimento do projeto da *Shell* Orion *Data Mining Engine* tornam a ferramenta mais completa e capaz de ser utilizada em pesquisas acadêmicas, bem como na busca de conhecimento por empresas que atuam no mercado de trabalho, sendo mais competitiva para com as ferramentas já existentes na área.

A *Shell* Orion é composta pelas tarefas de associação com o algoritmo *Apriori* (CASAGRANDE, 2005), classificação com os algoritmos ID3 (PELEGRIN, 2005) e CART (RAIMUNDO, 2007) e clusterização com *Kohonen* (BORTOLOTTO, 2007), *K-means* (MARTINS, 2007), *Gustafson-Kessel* (CASSETTARI JÚNIOR, 2008) e *Gath-Geva* (PEREGO, 2009). Porém, como esta ferramenta encontra-se em desenvolvimento, novas tarefas, métodos, algoritmos e funcionalidades estão constantemente sendo acrescentadas. No caso da pesquisa aqui apresentada, realizou-se a inserção no módulo de classificação de um algoritmo considerado importante, o C4.5.

O desenvolvimento do algoritmo C4.5 na *Shell* Orion foi baseado no livro intitulado *C4.5: Programs for Machine Learning* de John Rosss Quinlan que foi publicado em 1993. Neste livro encontra-se a explicação das etapas executadas pelo algoritmo, bem como os cálculos estatísticos realizados. Assim, os fundamentos e exemplos demonstrados a seguir referenciam a forma como Quinlan concebeu e exemplificou seu algoritmo.

A idéia original do funcionamento do algoritmo C4.5 baseia-se no trabalho de Hoveland e Hunt durante a década de 1950, consequentemente dando origem a um dos primeiros livros do assunto, *Experiments in Induction* (HUNT; MARIN; STONE,

1966 apud QUINLAN, 1993, tradução nossa), e influenciando também trabalhos como o do algoritmo CART (BREIMAN et al, 1984 apud QUINLAN, 1993, tradução nossa) e ID3. Este último também produzido por Quinlan em meados dos anos 80.

O desenvolvimento do módulo do algoritmo C4.5 na *Shell* Orion foi realizado inicialmente por sua modelagem matemática, seguido da implementação deste. Utilizando-se uma base de dados de exemplo foi possível realizar os cálculos existentes no algoritmo, comparando-se os resultados gerados com os provenientes da ferramenta original disponibilizada por Quinlan.

7.1 BASE DE DADOS

A base de dados utilizada para a realização dos testes da implementação do C4.5 na *Shell* Orion referem-se a informações das flores do família *Iridaceas*, presente as espécies *Iris-virginica*, *Iris-versicolor* e *Iris-setosa* (Figura 31). Estes dados estão disponíveis juntamente com a distribuição do algoritmo desenvolvido por Quinlan. Contendo 150 registros, a base possui dados das características físicas das flores e a qual espécie cada registro pertence, distribuídos em 50 registros para cada espécie. Os atributos da base são: largura da sépala; comprimento da sépala; largura da pétala; comprimento da pétala; nome da espécie.

Figura 31. Flores *Iris-virginica*, *Iris-versicolor* e *Iris-setosa*

Segundo Zanusso (2002) as espécies existentes da família *Iridaceas* dividem-se em mais de mil exemplares, sendo, desta forma, possível realizar a partir dos atributos com informações sobre as sépalas e pétalas a classificação dos dados em busca de se conhecer a qual espécie a flor pertence.

7.2 METODOLOGIA

O desenvolvimento do algoritmo C4.5 na *Shell* Orion foi realizado a partir do levantamento bibliográfico sobre o assunto, sendo seguido da modelagem matemática, implementação e testes. Após a etapa de testes com o algoritmo implementado na *Shell* Orion, foi avaliado a sua execução entre a ferramenta Weka e a implementação original do algoritmo C4.5, desenvolvida por Quinlan, para avaliação da exatidão dos resultados obtidos e tempo de processamento envolvidos.

Após a pesquisa bibliográfica de diferentes materiais sobre o algoritmo desenvolvido, obteve-se o conhecimento necessário para a demonstração das etapas realizadas pelo mesmo. Salienta-se que há uma escassez de trabalhos que utilizam e descrevam de forma detalhada os passos da sua execução. Além disso, muitos dos trabalhos encontrados referem-se apenas a aplicação do DM por meio de diferentes ferramentas disponíveis no mercado.

Mediante isso, a fim de atingir os objetivos desta pesquisa, delinearam-se as etapas metodológicas de: modelagem do módulo por meio de *Unified Modeling Language*[31] (UML); demonstração da modelagem matemática do algoritmo C4.5; implementação e testes do algoritmo C4.5 na *Shell* Orion.

[31] É uma linguagem expressa graficamente com a finalidade de demonstrar arquiteturas e aspectos comportamentais de software (JÜRJENS, 2005, tradução nossa).

7.2.1 Modelagem do Módulo de Classificação pelo Algoritmo C4.5

O desenvolvimento teve início pela realização da modelagem das etapas executadas pelo módulo do algoritmo C4.5 utilizando UML, por meio da ferramenta gratuita JUDE[32], objetivando a melhor compreensão dos processos executados.

Os diagramas desenvolvidos foram de: caso de uso, atividades e sequências.

O diagrama de caso de uso demonstra de que forma o usuário utiliza o módulo do algoritmo C4.5 na *Shell* Orion. Para tanto, dois passos precisam ser realizados para obtenção do resultado (Figura 32):

a) **1º passo**: usuário informa os parâmetros para em seguida requisitar a execução do algoritmo;

b) **2º passo**: é realizada a execução do C4.5 e retorna-se os resultados para o usuário.

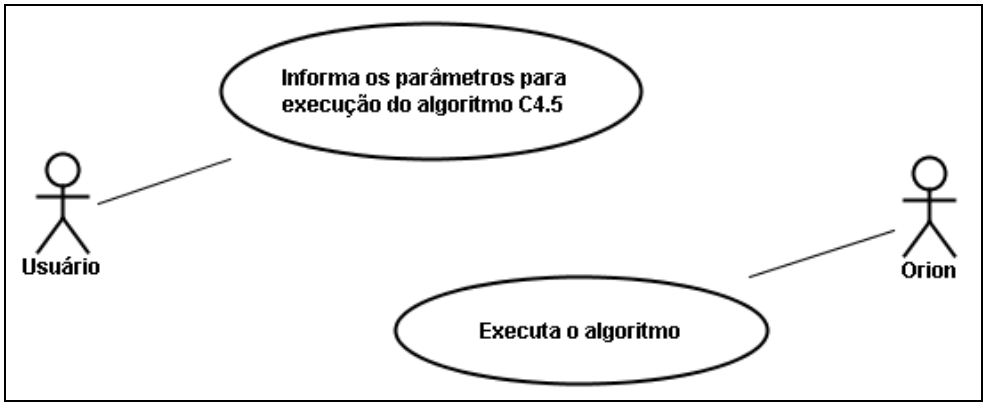

Figura 32. Diagrama de caso de uso do módulo do algoritmo C4.5

Estes passos são realizados por ação tanto do usuário quanto do sistema, sendo que inicialmente o usuário deverá informar os parâmetros para em seguida requisitar a execução do algoritmo. O sistema então irá processar a requisição para

[32] Disponível para download no site http://jude.change-vision.com/jude-web/download/index.html

retornar os resultados obtidos, proporcionando uma forma de visualização para o usuário.

As atividades presentes no módulo do algoritmo C4.5 estão divididas entre as realizadas pelo usuário e as realizadas pela *Shell* Orion, onde o usuário estará apto a informar os parâmetros para execução do algoritmo, solicitar sua execução e visualizar os resultados. A atividade única a ser realizada pela *Shell* Orion será a de executar o algoritmo após a solicitação feita pelo usuário, assim como pode ser visto na Figura 33.

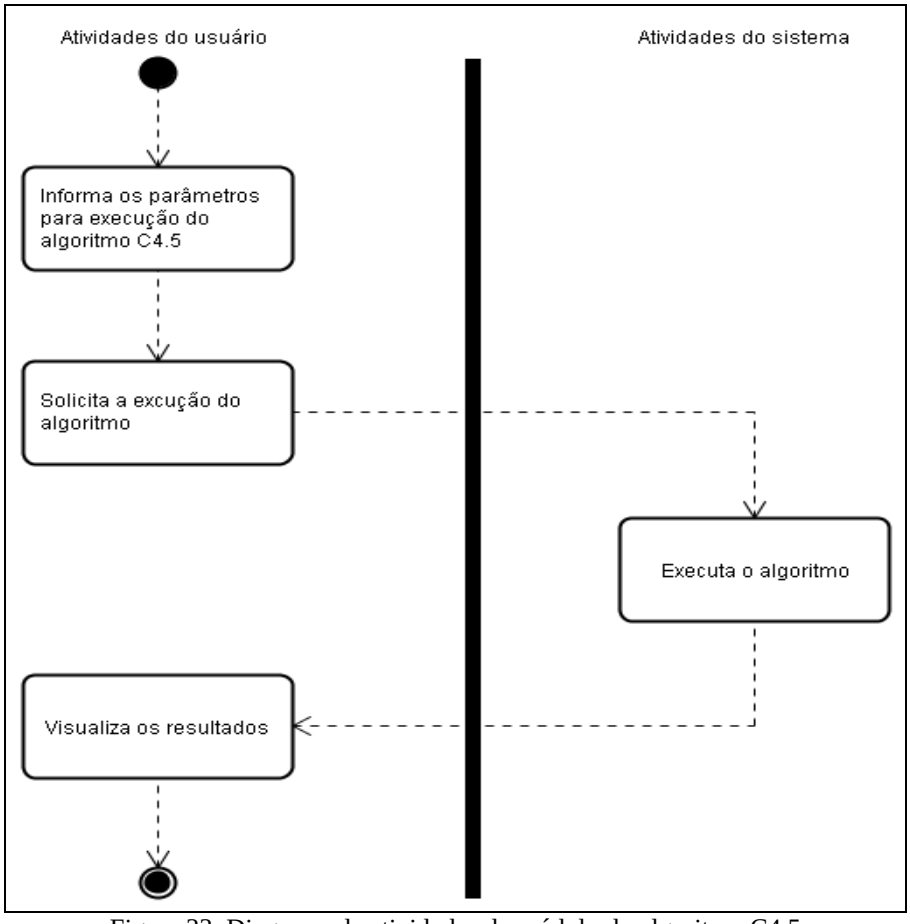

Figura 33. Diagrama de atividades do módulo do algoritmo C4.5

As ações executadas no módulo do algoritmo C4.5 estão dispostas inicialmente pelo processo de abrir o módulo, realizado pelo usuário, a execução do algoritmo é processada onde retorna-se os resultados obtidos, que finalmente são

disponibilizados ao usuário, como demonstra-se pelo diagrama de sequências na Figura 34.

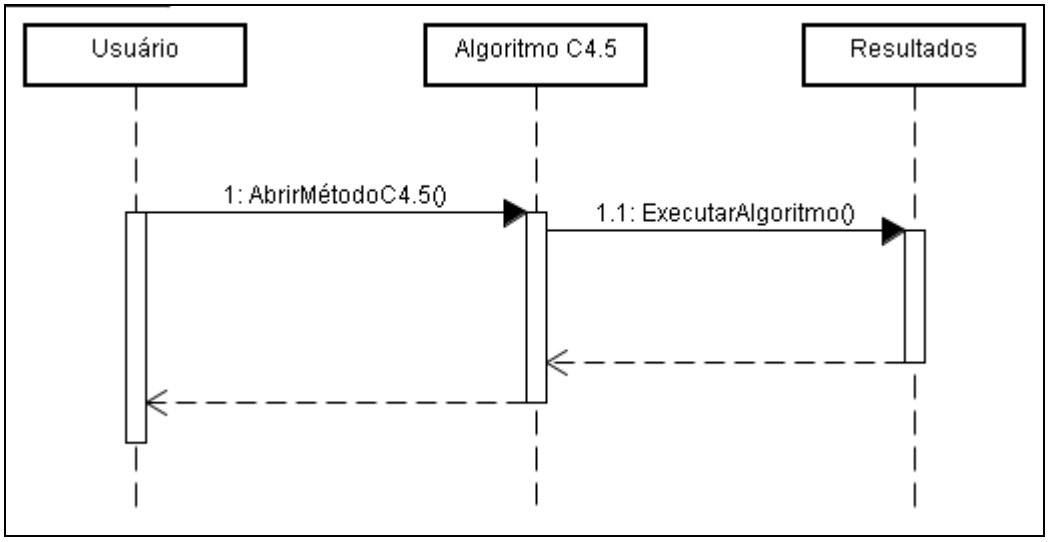

Figura 34. Diagrama de sequências do módulo do algoritmo C4.5

A modelagem do módulo tende a servir de ajuda principalmente para o caso da ferramenta vir a agregar outros algoritmos, pois assim que houver uma quantidade significativa de métodos disponíveis para o usuário utilizar, unificando-se suas modelagens ficará mais compreensível como separadamente cada método funciona.

7.2.2 Demonstração da Modelagem Matemática do Algoritmo C4.5

A execução do algoritmo C4.5 deve ser iniciada pela definição dos parâmetros, como foi comentado anteriormente. Dentre os parâmetros, o usuário deverá informar:

a) **tabela:** tabela da base de dados a ser utilizada para a classificação;

b) **atributo de saída:** coluna da tabela selecionada a ser demonstrado o resultado da classificação;

c) **método de cálculo:** forma do cálculo para seleção do teste para o nó da árvore, calculado pelo ganho de informação ou pela razão do ganho de informação. Utiliza-se a segunda opção por padrão;

d) **quantidade mínima de registros:** quantidade mínima de registros para a ramificação de um nó, sendo dois registros por padrão;

e) **nível de confiança (poda):** percentual de confiança para a realização da poda que será utilizado nos casos que ocorrerem erros acima do percentual indicado pelo usuário. No algoritmo C4.5 tem-se como padrão o nível de confiança de 25%.

Os valores informados por padrão são provenientes do algoritmo C4.5 desenvolvido por Quinlan, onde os valores apresentados são os fatores geradores do conhecimento obtido na execução, pois valores diferentes do padrão apresentados não constroem classificadores corretamente como o esperado.

Após as definições dos parâmetros para a execução do algoritmo, iniciam-se as etapas de forma recursiva para a classificação dos dados. Considerando-se um conjunto de dados T, sendo suas classes denotadas como $\{C_1, C_2, C_3, ..., C_j\}$, tem-se 3 possibilidades:

a) **T possui um ou mais casos pertencentes a classe C_j:** a árvore de decisão é uma folha identificada pela classe C_j.

b) **T não possui mais de um caso:** a árvore de decisão é neste ponto uma folha, onde a classe a ser utilizada nela é determinada por meio de informações, como por exemplo, a mais frequente no nó superior da folha;

c) **T contém casos pertencentes a mais de uma classe:** neste caso deseja-se dividir o conjunto de dados em partições baseadas em suas diferentes

classes. Utilizando um atributo em particular, obtem-se seus possíveis valores, sendo considerado um nó, onde terá um ramo para cada valor.

De forma a demonstrar os cálculos realizados para a construção da árvore, tem-se o método chamado ganho de informação, sendo descrito por meio da equação:

$$gain(T) = info(T) - info_x(T)$$

Onde:

- *info(T)* é definida como o valor da informação gerada pela classe do conjunto

$$info(T) = -\sum_{j=1}^{k} \frac{freq(C_j, T)}{|T|} \times \log_2 \left(\frac{freq(C_j, T)}{|T|} \right) bits$$

- $info_x(T)$ é o valor da informação obtida por um determinado atributo

$$info_x(T) = \sum_{i=1}^{n} \frac{|T_i|}{|T|} \times info(T_i)$$

Estes são os cálculos utilizados também pelo algoritmo ID3, diferenciando-se no C4.5 pela divisão do atributo escolhido para o nó. Atributos nominais terão um ramo para cada possível valor contido no conjunto, enquanto os atributos numéricos serão divididos em dois ramos. Isso ocorre por meio da divisão do valor do resultado do cálculo do ganho de informação pela informação gerada por um dos possíveis valores do atributo, chamado de razão do ganho de informação, definido pela seguinte equação:

$$gain\ ratio(X) = gain(X) / split\ info(X)$$

Onde *split info(X)* é definido como:

$$split\ info(X) = -\sum_{i=1}^{n} \frac{|T_i|}{|T|} \times \log_2 \left(\frac{|T_i|}{|T|} \right)$$

Exemplificando a utilização das equações, tem-se a base de dados com registros sobre as flores *Iridaceas*. Com 150 registros, e atributos contendo informações

sobre tamanho e largura, das pétalas e sépalas das flores, é possível realizar a classificação das mesmas em suas espécies: *Iris-setosa*, *Iris-versicolor* e *Iris-virginica*.

A demonstração dos cálculos realizados pelo algoritmo C4.5 implica na utilização de menos valores que o contido na base de dados, pois os cálculos por meio de todos os registros da base tornam-se extensos e complexos. Desta forma, a fim de exemplificar as etapas realizadas pelo algoritmo C4.5, faz-se uso dos dados presentes na Tabela 11.

Tabela 11. Dados exemplares referente as flores *Iridaceas* para demonstração do algoritmo C4.5

Comprimento da sépala	Largura da sépala	Comprimento da pétala	Largura da pétala	Espécie da flor
5,20	3,20	0,60	4,90	Iris-setosa
5,20	3,20	0,60	4,90	Iris-setosa
4,60	3,70	0,60	4,90	Iris-setosa
5,20	3,20	3,00	5,00	Iris-versicolor
4,80	3,40	0,70	4,90	Iris-versicolor
4,80	3,40	1,70	4,60	Iris-versicolor
4,60	3,70	0,70	4,90	Iris-versicolor
5,20	3,20	2,70	4,90	Iris-virginica
4,80	3,40	0,70	5,00	Iris-virginica
4,80	3,40	2,70	4,90	Iris-virginica
4,60	3,70	2,70	4,90	Iris-virginica
4,60	3,70	0,70	5,00	Iris-virginica

Inicialmente define-se qual o atributo de saída (ou classe), neste caso *espécie da flor*. Para cada atributo do conjunto, menos a classe, é realizado o teste a fim de verificar o ganho obtido, sendo ele considerado o nó da árvore. Tendo o ganho como a diferença entre o valor da informação para a classe em todo o conjunto de dados e o valor para o atributo testado, tem-se:

- *T = espécie da flor* (classe),
- 3 registros pertencendo a classe *Iris-setosa*,
- 4 registros pertencendo a classe *Iris-versicolor*,
- 5 registros pertencendo a classe *Iris-virginica*,

- Total de registros = 12.

$$info(T) = -\sum_{j=1}^{k} \frac{freq(C_j, T)}{|T|} \times \log_2\left(\frac{freq(C_j, T)}{|T|}\right) bits$$

$$info(T) = -\frac{3}{12} \times \log_2\left(\frac{3}{12}\right) - \frac{4}{12} \times \log_2\left(\frac{4}{12}\right) - \frac{5}{12} \times \log_2\left(\frac{5}{12}\right) = 1{,}550\, bits$$

Iniciando-se a busca pelo atributo com maior razão do ganho de informação dentre os contidos na Tabela 11, além da classe, tem-se:

- *x = comprimento da pétala*, nesta etapa é então calculado o valor da informação para cada possível valor existente no atributo,
- Total de registros = 12,
- atributo = *comprimento de pétala*,
- 3 registros com valor 0,60, sendo estes para *Iris-setosa*,
- 4 registros com valor 0,70, sendo 2 para *Iris-versicolor* e outros 2 para *Iris-virginica*,
- 1 registro com valor 1,70, sendo este para *Iris-versicolor*,
- 3 registros com valor 2,70, sendo estes para *Iris-virginica,*
- 1 registro com valor 3,00, sendo este para *Iris-versicolor*.

$$info_x(T) = \sum_{i=1}^{n} \frac{|T_i|}{|T|} \times info(T_i)$$

$$info_x\left(\begin{matrix}comprimento\\pétala\end{matrix}\right)=\frac{3}{12}\times\left(-\frac{3}{12}\times\log_2\left(\frac{3}{12}\right)-\frac{0}{12}\times\log_2\left(\frac{0}{12}\right)-\frac{0}{12}\times\log_2\left(\frac{0}{12}\right)\right)+$$
$$\frac{4}{12}\times\left(-\frac{0}{12}\times\log_2\left(\frac{0}{12}\right)-\frac{2}{12}\times\log_2\left(\frac{2}{12}\right)-\frac{2}{12}\times\log_2\left(\frac{2}{12}\right)\right)+$$
$$\frac{1}{12}\times\left(-\frac{0}{12}\times\log_2\left(\frac{0}{12}\right)-\frac{1}{12}\times\log_2\left(\frac{1}{12}\right)-\frac{0}{12}\times\log_2\left(\frac{0}{12}\right)\right)+$$
$$\frac{3}{12}\times\left(-\frac{0}{12}\times\log_2\left(\frac{0}{12}\right)-\frac{0}{12}\times\log_2\left(\frac{0}{12}\right)-\frac{3}{12}\times\log_2\left(\frac{3}{12}\right)\right)+$$
$$\frac{1}{12}\times\left(-\frac{0}{12}\times\log_2\left(\frac{0}{12}\right)-\frac{1}{12}\times\log_2\left(\frac{1}{12}\right)-\frac{0}{12}\times\log_2\left(\frac{0}{12}\right)\right)$$

$$info_x\left(\begin{matrix}comprimento\\pétala\end{matrix}\right)=0{,}587\,bits$$

$$gain(T)=info(T)-info_x(T)$$

$$gain\left(\begin{matrix}comprimento\\pétala\end{matrix}\right)=1{,}550-0{,}587=0{,}963\,bits$$

$$split\ info(X)=-\sum_{i=1}^{n}\frac{|T_i|}{|T|}\times\log_2\left(\frac{|T_i|}{|T|}\right)$$

$$split\ info(\begin{matrix}comprimento\\pétala\end{matrix})=(-3/12)\times\log_2(3/12)+$$
$$(-4/12)\times\log_2(4/12)+$$
$$(-1/12)\times\log_2(1/12)+$$
$$(-3/12)\times\log_2(3/12)+$$
$$(-1/12)\times\log_2(1/12)$$

$$split\ info(\begin{matrix}comprimento\\pétala\end{matrix})=2{,}130\,bits$$

$$gain\ ratio(X)=gain(X)/split\ info(X)$$

$$gain\ ratio\left(\begin{matrix}comprimento\\pétala\end{matrix}\right)=0{,}963/2{,}130=0{,}452\,bits$$

O cálculo é repetido para os demais atributos, onde neste ponto o *comprimento da pétala* possui a maior razão do ganho de informação, sendo então atribuído ao nó como sua partição. A árvore é então iniciada com o atributo *comprimento da pétala*, como pode ser visto na Figura 35.

Figura 35. Primeiro nível da árvore de decisão da base de dados das flores *Iridaceas*

A escolha do nó é realizada por um atributo numérico, como mostrado. A divisão é então realizada em dois ramos, resultando em um dos possíveis valores do atributo escolhido para o nó tendo como teste: A <= Z para o ramo da esquerda e A > Z para o ramo da direita, onde A refere-se ao atributo e Z a um de seus valores.

De maneira semelhante à busca do nó para utilizar como particionamento, para a escolha do valor da partição, como neste caso o atributo é numérico, é realizado o cálculo entre os valores existentes nos registros para o atributo *comprimento da pétala*. Ordenando-os de forma ascendente, o algoritmo C4.5 realiza uma divisão entre dois grupos, tentando encontrar o valor com maior razão do ganho de informação, sendo:

- *x = comprimento da pétala* com valor 0,60,
- 3 registros possuem o valor <= 0,60, sendo estes para *Iris-setosa*,
- 9 registros são os demais, > 0,60, sendo 4 registros para *Iris-versicolor* e 5 para *Iris-virginica*.

$$info_x(0{,}60) = \frac{3}{12} \times \left(-\frac{3}{12} \times \log_2\left(\frac{3}{12}\right) - \frac{0}{12} \times \log_2\left(\frac{0}{12}\right) - \frac{0}{12} \times \log_2\left(\frac{0}{12}\right) \right) + \frac{9}{12} \times \left(-\frac{0}{12} \times \log_2\left(\frac{0}{12}\right) - \frac{4}{12} \times \log_2\left(\frac{4}{12}\right) - \frac{5}{12} \times \log_2\left(\frac{5}{12}\right) \right)$$

$$info_x(0{,}60) = 0{,}916\, bits$$

$$gain(0{,}60) = 1{,}550 - 0{,}916 = 0{,}634\, bits$$

$$split\ info(0{,}60) = (-3/12) \times \log_2(3/12) + (-9/12) \times \log_2(9/12) = 0{,}810\, bits$$

$$gain\ ratio(0{,}60) = 0{,}634 / 0{,}810 = 0{,}560\, bits$$

Em seguida realiza-se os cálculos para o próximo valor existente no atributo *comprimento da pétala*, sendo neste caso o valor de 0,60, que é confirmado como a maior razão do ganho de informação entre os demais valores. Atribui-se às ramificações do nó *comprimento de pétala* o valor 0,60 (Figura 36).

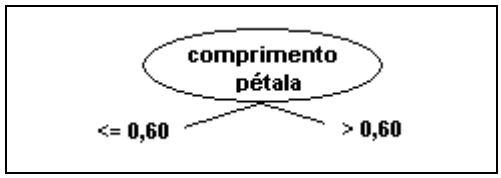

Figura 36. Primeiro nível ramificado da árvore de decisão da base de dados das flores *Iridaceas*

A ramificação do teste <= 0,60 possui todos os registros pertencentes a mesma classe, criando-se uma folha a que se atribui a classe *Iris-setosa*. A ramificação do teste > 0,60 busca por mais partições possíveis de serem realizadas. Mais de uma classe em comum é encontrada nos registros, iniciando-se os cálculos onde:

- x = *largura da pétala*,
- Total de registros = 9,
- 1 registro com valor 4,60, sendo este para *Iris-versicolor*,
- 5 registros com valor 4,90, sendo 2 para *Iris-versicolor* e outros 3 para *Iris-virginica*,
- 3 registros com valor 5,00, sendo 1 para *Iris-versicolor* e outros 2 para *Iris-virginica*.

$$info_x\left(\begin{matrix}largura\\ pétala\end{matrix}\right) = \frac{1}{9}\times\left(-\frac{0}{9}\times\log_2\left(\frac{0}{9}\right)-\frac{1}{9}\times\log_2\left(\frac{1}{9}\right)-\frac{0}{9}\times\log_2\left(\frac{0}{9}\right)\right)+$$
$$\frac{5}{9}\times\left(-\frac{0}{9}\times\log_2\left(\frac{0}{9}\right)-\frac{2}{9}\times\log_2\left(\frac{2}{9}\right)-\frac{3}{9}\times\log_2\left(\frac{3}{9}\right)\right)+$$
$$\frac{3}{9}\times\left(-\frac{0}{9}\times\log_2\left(\frac{0}{9}\right)-\frac{1}{9}\times\log_2\left(\frac{1}{9}\right)-\frac{2}{9}\times\log_2\left(\frac{2}{9}\right)\right)$$

$$info_x\left(\begin{matrix}largura\\ pétala\end{matrix}\right) = 0{,}878\,bits$$

$$gain\left(\begin{matrix}largura\\pétala\end{matrix}\right)=1{,}550-0{,}878=0{,}672\,bits$$

$$split\ info(\begin{matrix}largura\\pétala\end{matrix}) = (-1/9)\times\log_2(1/9)+ (-5/9)\times\log_2(5/9)+ (-3/9)\times\log_2(3/9)=1{,}350\,bits$$

$$gain\ ratio\left(\begin{matrix}largura\\pétala\end{matrix}\right)=0{,}672/1{,}350=0{,}498\,bits$$

Realizando os cálculos com os demais atributos, *largura da pétala* é o que possui a maior razão do ganho de informação. A Figura 37 demonstra como encontra-se a árvore em construção dos resultados obtidos. Em seguida os valores do atributo *largura da pétala* são analisados a fim de identificar qual será utilizado para teste na ramificação do nó.

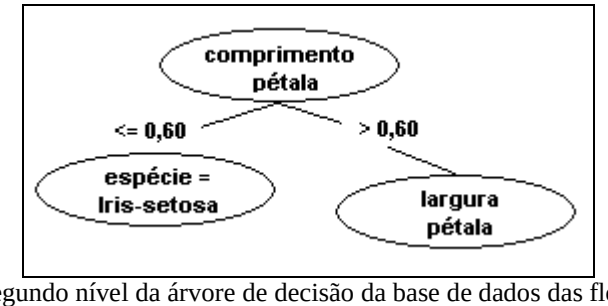

Figura 37. Segundo nível da árvore de decisão da base de dados das flores *Iridaceas*

Considerando:

- *x* = *largura da pétala* com valor 4,90,
- 6 registros possuem valor <= 4,90, sendo 3 registros para *Iris-versicolor* e 3 para *Iris-virginica*;
- 3 registros são os demais, > 4,90, sendo 1 registro para *Iris-versicolor* e 2 para *Iris-virginica*.

$$info_x(4{,}90)=\frac{6}{9}\times\left(-\frac{0}{9}\times\log_2\left(\frac{0}{9}\right)-\frac{3}{9}\times\log_2\left(\frac{3}{9}\right)-\frac{3}{9}\times\log_2\left(\frac{3}{9}\right)\right)+ \frac{3}{9}\times\left(-\frac{0}{9}\times\log_2\left(\frac{0}{9}\right)-\frac{1}{9}\times\log_2\left(\frac{1}{9}\right)-\frac{2}{9}\times\log_2\left(\frac{2}{9}\right)\right)$$

$$info_x(4{,}90)=0{,}982\,bits$$

$$gain(4{,}90) = 1{,}550 - 0{,}982 = 0{,}568\,bits$$

$$split\ info(4{,}90) = (-6/9) \times \log_2(6/9) + (-3/9) \times \log_2(3/9) = 0{,}920\,bits$$

$$gain\ ratio(4{,}90) = 0{,}568 / 0{,}920 = 0{,}617\,bits$$

Dentre os demais valores do atributo *largura da pétala*, o valor de 4,90 é o que contém maior razão do ganho de informação, sendo este então associado ao nó para utilizar como teste da ramificação. Assim como demonstra a Figura 38, a árvore em construção estaria no segundo nível com a sua ramificação já destacada.

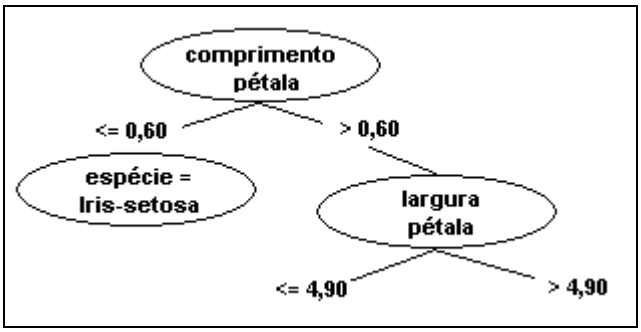

Figura 38. Segundo nível ramificado da árvore de decisão da base de dados das flores *Iridaceas*

Os cálculos são novamente repetidos nas ramificações obtidas no nó apresentado pelo atributo *largura da pétala*. Ao ser realizada a análise para a ramificação contendo o teste > 4,90, será identificado que existem duas classes ainda nos registros deste nó, sendo três registros para a classe *Iris-virginica* e um para a classe *Iris-versicolor*. Nesta ocasião é utilizada a quantidade mínima de registros que um nó deve conter que, ao ser realizada a ramificação do nó subsequente à 4,90, apresentaria então uma folha com apenas um registro, sendo neste caso atribuído os resultados de três registros classificados corretamente e um erroneamente.

Este caso é discutido por Quinlan (1993), argumentando que ao ser realizada tal ramificação obtendo-se uma folha com apenas um registro, não há conhecimento algum. Neste caso, a árvore construída apresentaria apenas uma disposição dos valores

encontrados na base de dados, de forma a construir regras simples em virtude de um agrupamento de valores dos atributos analisados.

A busca por uma nova partição é realizada na ramificação <= 4,90, onde:

- *x = comprimento da pétala*,

- Total de registros = 7,

- 2 registros com valor 0,70, sendo estes para *Iris-versicolor*,

- 1 registro com valor 1,70, sendo estes para *Iris-versicolor*,

- 3 registros com valor 2,70, sendo estes para *Iris-virginica*.

$$info_x\left(\begin{matrix}comprimento\\pétala\end{matrix}\right)=\frac{2}{7}\times\left(-\frac{0}{7}\times\log_2\left(\frac{0}{7}\right)-\frac{2}{7}\times\log_2\left(\frac{2}{7}\right)-\frac{0}{7}\times\log_2\left(\frac{0}{7}\right)\right)+$$
$$\frac{1}{7}\times\left(-\frac{0}{7}\times\log_2\left(\frac{0}{7}\right)-\frac{1}{7}\times\log_2\left(\frac{1}{7}\right)-\frac{0}{7}\times\log_2\left(\frac{0}{7}\right)\right)+$$
$$\frac{3}{7}\times\left(-\frac{0}{7}\times\log_2\left(\frac{0}{7}\right)-\frac{0}{7}\times\log_2\left(\frac{0}{7}\right)-\frac{3}{7}\times\log_2\left(\frac{3}{7}\right)\right)$$

$$info_x\left(\begin{matrix}comprimento\\pétala\end{matrix}\right)=0{,}429\,bits$$

$$gain\left(\begin{matrix}comprimento\\pétala\end{matrix}\right)=1{,}550-0{,}429=1{,}121\,bits$$

$$split\ info(\begin{matrix}comprimento\\pétala\end{matrix})=(-2/7)\times\log_2(2/7)+$$
$$(-1/7)\times\log_2(1/7)+$$
$$(-3/7)\times\log_2(3/7)=1{,}440\,bits$$

$$gain\ ratio\left(\begin{matrix}comprimento\\pétala\end{matrix}\right)=1{,}121/1{,}440=0{,}778\,bits$$

O terceiro e último nó da árvore é especificado pelo atributo *comprimento da pétala*, pois este obteve maior razão do ganho de informação. Nesta etapa, a árvore encontra-se como demonstrado na Figura 39.

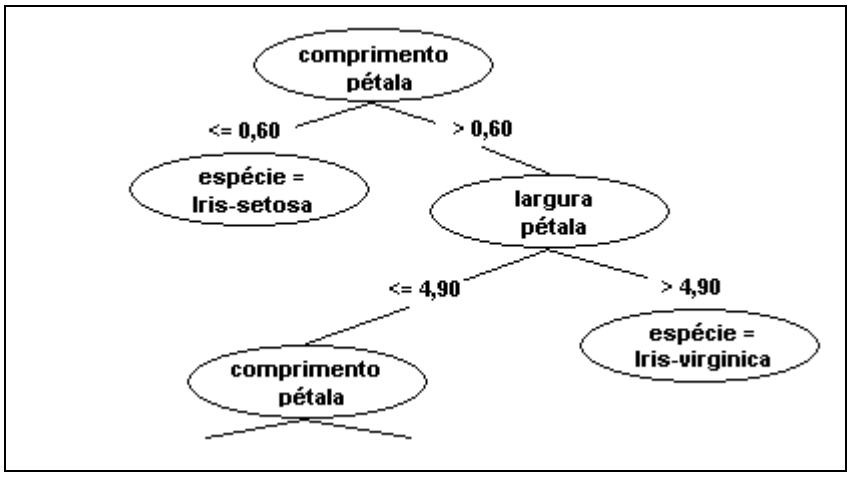

Figura 39. Terceiro nível da árvore de decisão da base de dados das flores *Iridaceas*

Considerando os valores abaixo calcula-se o valor utilizado para teste na ramificação do nó anterior:

- x = *comprimento da pétala* com valor 1,70,

- 3 registros possuem valor <= 1,70, sendo estes para *Iris-versicolor*;

- 3 registros são os demais, > 1,70, sendo este para *Iris-virginica*.

$$info_x(1{,}70) = \frac{3}{6} \times \left(-\frac{0}{6} \times \log_2\left(\frac{0}{6}\right) - \frac{3}{6} \times \log_2\left(\frac{3}{6}\right) - \frac{0}{6} \times \log_2\left(\frac{0}{6}\right) \right) + \frac{3}{6} \times \left(-\frac{0}{6} \times \log_2\left(\frac{0}{6}\right) - \frac{0}{6} \times \log_2\left(\frac{0}{6}\right) - \frac{3}{6} \times \log_2\left(\frac{3}{6}\right) \right)$$

$$info_x(1{,}70) = 0{,}500\,bits$$

$$gain(1{,}70) = 1{,}550 - 0{,}500 = 1{,}050\,bits$$

$$split\ info(1{,}70) = (-3/6) \times \log_2(3/6) + (-3/6) \times \log_2(3/6) = 1{,}000\,bits$$

$$gain\ ratio(1{,}70) = 1{,}050 / 1{,}000 = 1{,}050\,bits$$

O valor de 1,70 é atribuído como teste da ramificação do nó *comprimento de pétala* no terceiro nível da árvore. Identificando-se a disposição das classes de forma total para cada ramificação, sendo três registros para cada, onde todos possuem a

mesma classe, a árvore não necessita buscar novas partições. Neste ponto a sua construção é finalizada. A árvore construída contém quatro folhas e três nós (Figura 40).

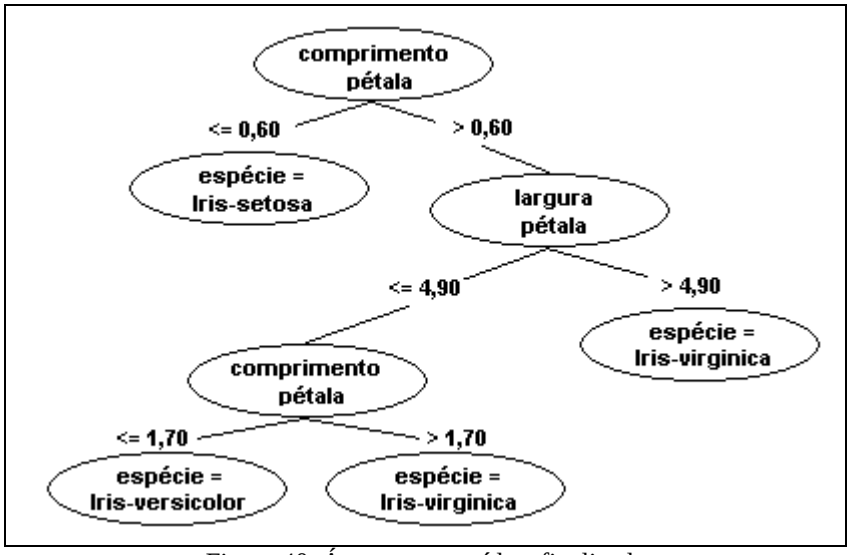

Figura 40. Árvore construída e finalizada

Ao final da construção da árvore o algoritmo C4.5 realiza a sua poda. Cada nó da árvore é avaliado com a quantidade de registros e de erros cometidos pelo classificador. No exemplo apenas o nó de segundo nível, com o atributo *largura da pétala*, conteve erro. Dentre os nove registros identificados neste nó, um destes não está corretamente classificado. Utilizando a equação obtem-se a taxa de erro:

$$U_{CF}(E,N)$$

$$U_{25\%}(1,9) = \left(9 * (0,25)^2\right) - 1 = -0,4375$$

A taxa de erro da raiz é calculada em seguida:

$$U_{CF}(E,N)$$

$$U_{25\%}(1,12) = \left(12 * (0,25)^2\right) - 1 = -0,25$$

A base utilizada para os testes demonstrados não ultrapassa o nível de confiabilidade padrão que seria de 25%, pois a taxa de erro do nó que contém registros que não pertencem a classe indicada são menores que a do nó avaliado.

A partir da compreensão destas etapas realizadas por meio de cálculos matemáticos foi possível dar-se início a implementação do algoritmo C4.5 na *Shell* Orion.

7.2.3 Implementação e Testes do Módulo de Classificação por meio do Algoritmo C4.5

A *Shell* Orion vem sendo desenvolvida na linguagem Java com a ferramenta gratuita NetBeans[33], sendo utilizada a versão 6.5.1 para a implementação do algoritmo C4.5.

No ano de 1993 John Ross Quinlan publicou a primeira versão do algoritmo no livro *C4.5: Programs for Machine Learning*. Nesta pesquisa a implementação do C4.5 foi realizada baseando-se na oitava versão deste algoritmo publicada pelo autor. Esta versão encontra-se disponível no *site http://www.rulequest.com/Personal*.

Na *Shell* Orion inicialmente realiza-se a conexão com a base de dados, sendo nesta pesquisa utilizada a conexão com o banco de dados SQL Anywhere 7.0[34] da Sybase, sendo que a empresa disponibiliza versões gratuitas de testes para usuários cadastrados no site, geralmente a fim de pesquisas científicas, projetos empresariais ao uso de novas ferramentas, entre outras finalidades.

Após a etapa de conexão, pode ser realizado o uso dos métodos das tarefas disponíveis, sendo desta forma possível iniciar a utilização do algoritmo C4.5 também. O acesso ao módulo deste é realizado pelo menu da ferramenta, onde navegando-se até *Data Mining* > *Classificação* > *C4.5* (Figura 41), será disponibilizado ao usuário a tela para inserção dos parâmetros de seu uso e demonstração dos resultados obtidos.

[33] Disponível para *download* em *http://www.netbeans.org*
[34] Disponível em http://www.sybase.com

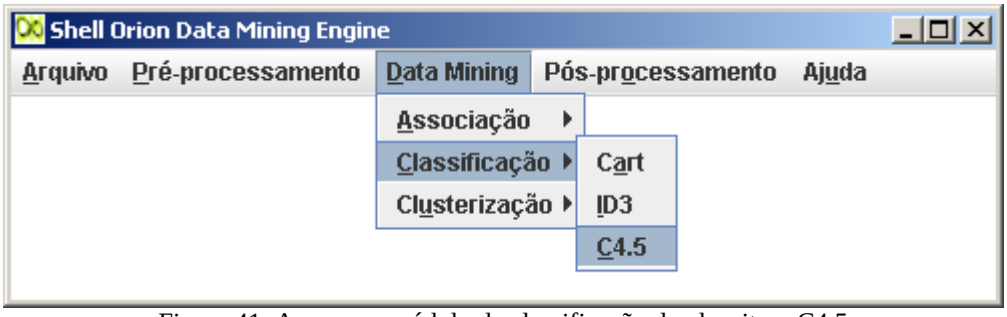

Figura 41. Acesso ao módulo de classificação do algoritmo C4.5

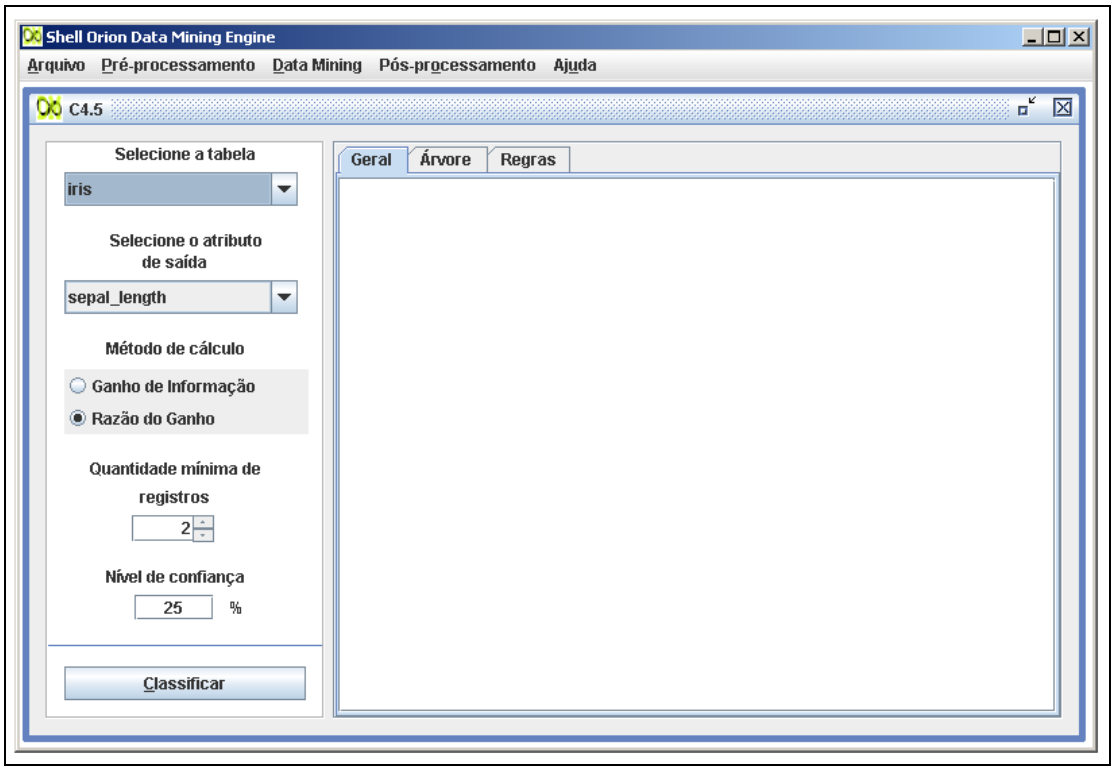

Figura 42. Módulo do algoritmo C4.5 na *Shell* Orion

A execução da tarefa de classificação pelo algoritmo C4.5 depende dos parâmetros que o usuário deverá informar, como pode ser visto na Figura 42, dentre estes:

a) **tabela:** tabela onde a ferramenta deverá realizar a busca dos dados para a execução da classificação;

b) **atributo de saída:** informação gerada ao final da classificação, demonstrando o que o usuário espera obter como consequente das regas;

c) **método de cálculo:** método pelo qual o usuário deseja que se obtenha os valores de ganho de informação para os atributos da tabela, sendo possível utilizar o ganho de informação ou a razão do ganho de informação;

d) **quantidade mínima de registros:** demonstra quantos registros o usuário deseja que o algoritmo utilize para ser definido como uma regra final, ou seja, quantos registros no mínimo deverão conter numa folha para poder ser considerado como folha. Poderá ser utilizado para verificar a diferença entre árvores que possuem alguma taxa de erro baixa em uma folha, onde ao ser aumentado o valor padrão utilizado pelo algoritmo, poderá ser criado uma nova árvore com pequenas diferenças, mas de pequena importância ao classificador;

e) **nível de confiança:** utilizado para informar o percentual de confiança gerado por uma regra, sendo neste caso necessário que o algoritmo tente podar a árvore final, a fim de diminuir a taxa de erro informada na regra.

Os resultados no módulo do algoritmo C4.5 podem ser visualizados na forma de regras e de árvore de decisão. Nas Figuras 43 e 44 é possível verificar como é disponibilizado ao usuário a análise dos resultados.

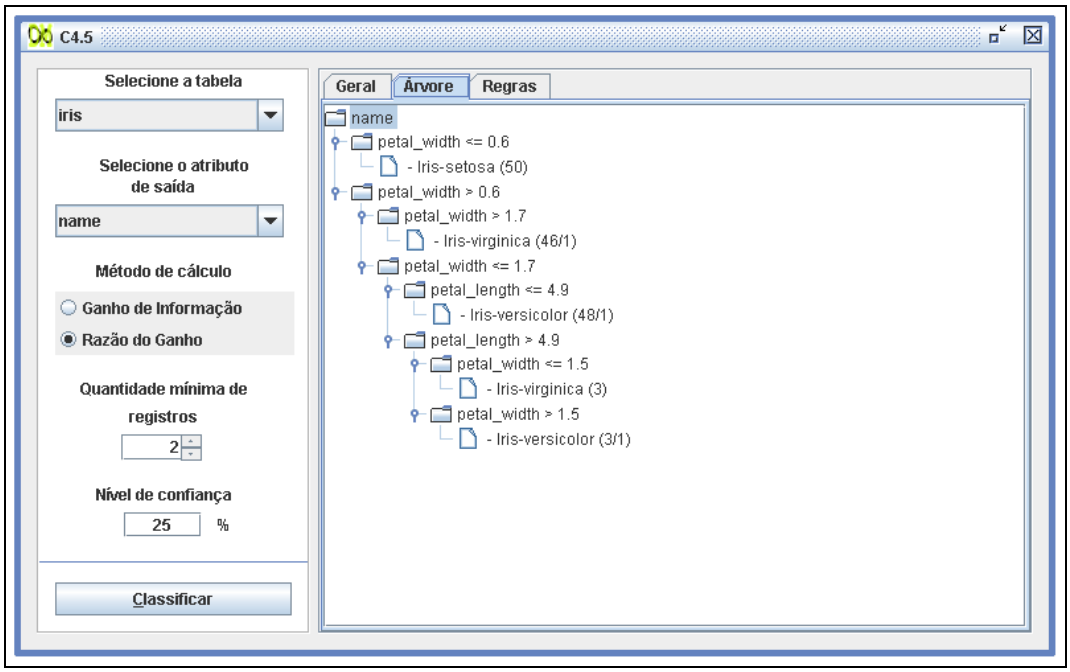

Figura 43. Resultados do algoritmo C4.5 visualizados por meio de árvore de decisão

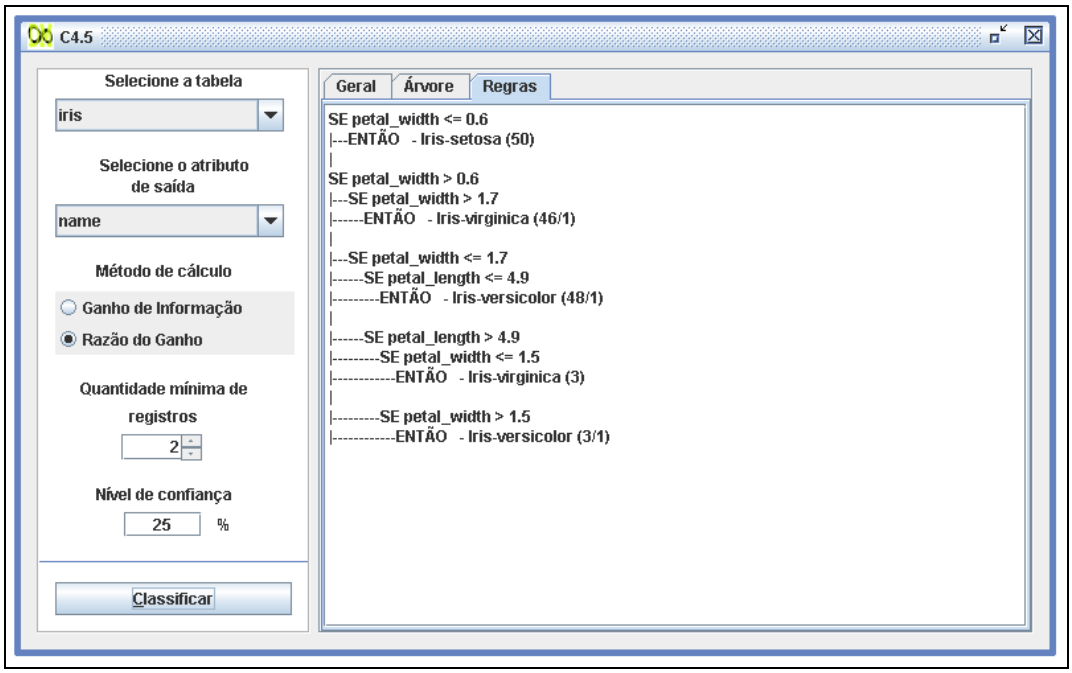

Figura 44. Resultados do algoritmo C4.5 visualizados por meio de regras

A implementação do algoritmo C4.5 obteve êxito ao ser testada e comparada com outras ferramentas disponíveis no mercado. A seguir será apresentado os resultados obtidos e a discussão destes.

7.3 RESULTADOS OBTIDOS

A finalização da implementação do algoritmo C4.5 na *Shell* Orion *Data Mining Engine* teve como etapa posterior a realização de testes e análise de seu desempenho quanto ao *hardware* utilizado, sendo também analisado os resultados pela Orion com aqueles de diferentes ferramentas e algoritmos.

A configuração do hardware utilizado para tais testes de desempenho em análise de tempo e registros foi um microcomputador com sistema operacional *Windows* XP *Professional* SP2, processador Intel 2,66 Ghz com 1 Gb de memória RAM.

A Tabela 12 demonstra alguns dos resultados obtidos na *Shell* Orion em função da quantidade de registros e o tempo utilizado para a classificação.

Tabela 12. Análise de resultado de processamento do algoritmo C4.5 na *Shell* Orion

Registros	Atributos	Início	Fim	Tempo
14	5	04:58:19.421	04:58:19.437	16 ms
150	15	05:03:00.562	05:03:00.593	31 ms
2514	30	05:05:23.093	05:05:24.937	1.844 ms
2800	30	05:08:18.453	05:08:26.421	7.968 ms

Analisando também as ferramentas C4.5, disponibilizada por Quinlan, e a ferramenta Weka, pode-se verificar nas Tabelas 13 e 14, que o tempo de processamento entre elas é parecido, sendo que em relação a Orion, a diferença apresenta-se um pouco mais elevada. As ferramentas analisadas não possuem uma saída informando a hora inicial e final do processamento, mas possuem o tempo envolvido, como apresenta-se a seguir.

Tabela 13. Análise de resultado de processamento do algoritmo C4.5 distribuído pelo autor

Registros	Atributos	Início	Fim	Tempo
14	5	-	-	15 ms
150	15	-	-	15 ms
2514	30	-	-	125 ms
2800	30	-	-	250 ms

Tabela 14. Análise de resultado de processamento do algoritmo C4.5 na ferramenta Weka

Registros	Atributos	Início	Fim	Tempo
14	5	-	-	< 10 ms
150	15	-	-	< 10 ms
2514	30	-	-	90 ms
2800	30	-	-	220 ms

O processamento do algoritmo pelas ferramentas pode ser analisado conforme a Tabela 15. A quantidade de registros e de atributos nem sempre parece influenciar no tempo de execução, em vista de que o algoritmo possui diferentes tratamentos para atributos numéricos ou categóricos, ocasionando assim um aumento diferenciado no tempo de execução para as tabelas com trinta atributos por exemplo.

Tabela 15. Análise dos resultados de processamento nas ferramentas C4.5, Weka e *Shell* Orion

Ferramenta	Atributos / Registros	Tempo	Atributos / Registros	Tempo	Atributos / Registros	Tempo
C4.5	5 / 14	15 ms	15 / 150	15 ms	30 / 2800	250 ms
Weka	5 / 14	< 10 ms	15 / 150	< 10 ms	30 / 2800	220 ms
Shell Orion	5 / 14	16 ms	15 / 150	31 ms	30 / 2800	7.968 ms

O tempo utilizado para a classificação da base das flores da família *Iridaceas* foi considerado satisfatório quando comparado com outras ferramentas, como a Weka e o próprio algoritmo C4.5 distribuído pelo seu autor, onde em menos de um segundo as mesmas retornam os resultados obtidos.

A correta classificação na maioria dos registros utilizados para o teste também demonstrou o fato de como um classificador deve funcionar. As regras geradas obtiveram resultados que confirmam as espécies das flores as quais se distribuem nos registros contidos na base.

A avaliação dos resultados a partir dos processos realizados no algoritmo C4.5 pode ser complementada com a análise de desempenho do algoritmo, sendo realizado por meio de cálculos de média e desvio padrão obtidos em Rezende (2005). O cálculo da média, ou erro médio, é realizado pela Equação (6), onde:

$$mean(A) = \frac{1}{r}\sum_{i=1}^{r} err(h_i) \quad (6)$$

- *r* é o número de partições de um conjunto de dados utilizadas na análise,

- *err* é a taxa de erro dos registros analisados, obtido pela Equação (7), onde:

$$err(h) = \frac{1}{n}\sum_{i=1}^{n} \| y_i \neq h(x_i) \| \quad (7)$$

- *n* é o número de exemplos do conjunto de dados,

- y_i é a classe indicada no registro a ser classificado,

- $h(x_i)$ é o resultado retornado pelo classificador para o registro x_i,

- $\| y_i \neq h(x_i) \|$ retorna verdadeiro (1) caso o classificador indique a classe errada ao registro x_i ou falso (0) caso contrário.

Utilizando-se os dados contidos nas Tabelas 16, 17 e 18, são realizados então os cálculos para conhecer o erro médio do classificador gerado pelo algoritmo C4.5 na *Shell* Orion das flores da família *Iridaceas* baseado nos dados da Tabela 11.

Tabela 16. Primeira partição para análise do desempenho do algoritmo C4.5

Comprimento da sépala	Largura da sépala	Comprimento da pétala	Largura da pétala	Espécie da flor
5,40	3,90	1,30	0,40	Iris-setosa
5,70	3,80	1,70	0,30	Iris-setosa
6,30	3,30	4,70	1,60	Iris-versicolor
6,60	2,90	4,60	1,30	Iris-versicolor
5,70	2,80	4,10	1,30	Iris-versicolor
5,80	2,70	5,10	1,90	Iris-virginica
6,30	2,90	5,60	1,80	Iris-virginica

Tabela 17. Segunda partição para análise do desempenho do algoritmo C4.5

Comprimento da sépala	Largura da sépala	Comprimento da pétala	Largura da pétala	Espécie da flor
5,10	3,40	1,50	0,20	Iris-setosa
4,50	2,30	1,30	0,30	Iris-setosa
4,90	2,40	3,30	1,00	Iris-versicolor
5,20	2,70	3,90	1,40	Iris-versicolor
7,10	3,00	5,90	2,10	Iris-virginica
6,50	3,00	5,80	2,20	Iris-virginica
4,90	2,50	4,50	1,70	Iris-virginica

Tabela 18. Terceira partição para análise do desempenho do algoritmo C4.5

Comprimento da sépala	Largura da sépala	Comprimento da pétala	Largura da pétala	Espécie da flor
5,10	2,50	3,00	1,10	Iris-versicolor
6,30	3,30	6,00	2,50	Iris-virginica
6,40	3,20	4,50	1,50	Iris-versicolor
5,50	2,30	4,00	1,30	Iris-versicolor
5,90	3,00	5,10	1,80	Iris-virginica
4,90	3,00	1,40	0,20	Iris-setosa
4,40	3,20	1,30	0,20	Iris-setosa

Analisando as três partições, é realizado o cálculo do erro médio por:

- *n* é o total de registros da partição,
- calcula-se o erro da primeira partição, sendo que a expressão que verifica se o classificador realiza corretamente a classificação para os registros da primeira partição apresenta valor falso (0) em todos os registros,

$$err(h) = \frac{1}{7} \times (0+0+0+0+0+0+0) = 0$$

- calcula-se o erro da segunda partição, sendo que o último dos registros é classificado erroneamente, conforme o classificador gerado,

$$err(h) = \frac{1}{7} \times (0+0+0+0+0+0+1) = 0{,}143$$

- calcula-se por fim o erro da terceira partição.

$$err(h) = \frac{1}{7} \times (0+0+0+0+0+0+0) = 0$$

$$mean(A) = \frac{1}{3} \times (0+0{,}143+0) = 0{,}047$$

O erro médio que o classificador obteve é de 4,7% dos casos analisados, determinando assim uma estimativa de que ao ser utilizado em outro conjunto de dados, o erro médio deverá ser levado em conta dos resultados esperados com a classificação, podendo obter uma média de erros de 4,7% entre os novos casos classificados.

O resultado obtido com o erro médio é então utilizado para calcular o desvio padrão do classificador, onde por meio das Equações (8) e (9) tem-se:

$$var(A) = \frac{1}{r}\left[\frac{1}{r-1}\sum_{i=1}^{r}\left(err(h_i) - mean(A)\right)^2\right] \quad (8)$$

- *var(A)* é denominado variância das partições,

- *r* é o número de partições de um conjunto de dados utilizadas na análise,

- *err(h_i)* é o erro obtido em cada partição,

- *mean(A)* é o erro médio calculado anteriormente,

$$sd(A) = \sqrt{var(A)} \quad (9)$$

- *sd(A)* é denominado desvio padrão do classificador.

O cálculo da variança e do desvio padrão do classificador é descrito como segue:

- $r = 3$ partições,

- *mean(A)* = 0,047.

$$var(A) = \frac{1}{3}\left[\frac{1}{3-1}\times\left((0-0{,}047)^2 + (0{,}143-0{,}047)^2 + (0-0{,}047)^2\right)\right]$$

$$var(A) = \frac{1}{3}\left[\frac{1}{2}\times(0{,}002 + 0{,}009 + 0{,}002)\right]$$

$$var(A) = \frac{1}{3}\times 0{,}0089$$

$$var(A) = 0{,}0029$$

$$sd(A) = \sqrt{0{,}0029}$$

$$sd(A) = 0{,}0538$$

O desvio padrão do classificador analisado é de 5,38% acima ou abaixo do erro médio obtido anteriormente, de 4,7%, o classificador será considerado útil caso o

erro médio não ultrapasse 4,45% ou 4,95% com a utilização de novos registros classificados.

Analisando os valores obtidos pelo desempenho do algoritmo C4.5 na *Shell* Orion por meio dos dados contidos na Tabela 11, é possível verificar que o desvio padrão em virtude do erro médio obtido não demonstrou ser uma taxa significante de erros cometidos. Neste caso, com doze registros para a construção do classificador, ao ser utilizado outras três partições de dados, apenas um registro foi classificado erroneamente.

CONCLUSÃO

A aplicação do *data mining* para auxílio nas tomadas de decisões, empregadas nos mais diversos ramos, demonstra a importância do processo, dando também continuação aos estudos realizados e que ainda serão feitos nesta área. É importante destacar que cada vez mais as empresas direcionam o seu foco para este assunto.

A tarefa de classificação por meio do algoritmo C4.5 foi considerada de extrema importância para o conhecimento e aprimoramento de suas técnicas utilizadas, envolvendo desde o estudo sobre descoberta de conhecimento em bases de dados. A realização desta pesquisa veio motivar e concretizar novas pesquisas realizadas na área por ser um assunto tão amplo e de procura no mercado de trabalho.

A utilização do módulo implementado obteve êxito quanto a sua exatidão e operacionalidade em questão do tema abordado para realização de testes e exemplificações. A partir da aplicação do algoritmo C4.5 na tarefa de classificação no *data mining* realizado na base de dados, que contém informações sobre características físicas de flores da família *Iridaceas*, foi capaz de perceber como é construído um classificador e como são avaliados os dados contidos no conjunto. Os objetivos desta pesquisa foram alcançados devido ao estudo realizado sobre *data mining* e as funcionalidades existentes no algoritmo C4.5, além da compreensão obtida com a utilização de ferramentas que envolvem o assunto e com os resultados gerados por elas.

Além da classificação, os diferentes métodos utilizados no *data mining* são importantes para a comunidade científica, e também para os acadêmicos que venham empreender algum estudo na área. Dos métodos utilizados e presentes na *Shell* Orion, estes são uma pequena parte do que há disponível em fontes de pesquisa, sendo assim

possível aumentar a divulgação da ferramenta e do assunto a qual engloba, a fim de agregar mais e novos módulos à sua composição.

O comprometimento de futuros acadêmicos para o desenvolvimento da *Shell* Orion poderia contar com alguns exemplos de possíveis tarefas que venham a tornar a ferramenta mais fácil de usar e de ser implementada:

a) realizar estudo sobre os processos executados pelos algoritmos, até então implementados, a fim de padronizar cálculos e algumas etapas que são comuns entre eles;
b) implementar no módulo de classificação um algoritmo por meio do método de redes bayesianas;
c) pesquisar e desenvolver módulos de visualização dos resultados obtidos pelos algoritmos, tornando assim as futuras pesquisas mais simples de serem implementadas, em decorrência dos resultados serem apresentados de uma forma padrão de acordo com as saídas originadas, sejam elas em forma de gráfico, árvores ou as demais que venham a ser necessárias;
d) verificar sobre a evolução do algoritmo C4.5 que gerou um novo algoritmo chamado C5.0, realizando pesquisas sobre as diferenças encontradas entre estes e as estatísticas em relação a tempo e processamento entre eles, implementando também o C5.0 junto a *Shell* Orion.

REFERÊNCIAS

ALMENTERO, Bruno K; BAIÃO, Fernanda; MATTOSO, Marta. **Avaliação do desempenho de um algoritmo de árvore de decisão usando processamento paralelo e distribuição.** Relatório Técnico do Projeto Cluster Miner. Rio de Janeiro, 2004.

APERS, Peter; BOUZEGHOUB, Mokrane; GARDARIN Georges (Eds.). ***Advances in Database Technology:*** *procedings / EDBT '96, 5th International Conference on Extending Database Technology, Avignon, March 1996*. Alemanha: Springer, 1996.

BERRY, Michael J. A.; LINOFF, Gordon. ***Data mining techniques:*** *for marketing, sales, and customer relationship management.* Indiana: Wiley Publishing, 2004.

BORTOLOTTO, Leandro Sehnem. **O Método de Redes Neurais pelo Algoritmo Kohonen para Clustrerização na *Shell* Orion *Data Mining Engine*.** 90 f. Trabalho de Conclusão de Curso (Bacharel em Ciências da Computação) – Curso de Ciências da Computação, Universidade do Extremo Sul Catarinense, Criciúma, 2007.

CASAGRANDE, Diego Paz. **O Módulo da Tarefa de Associação pelo Algoritmo Apriori no Desenvolvimento da *Shell* de *Data Mining* Orion.** 79 f. Trabalho de Conclusão de Curso (Bacharel em Ciências da Computação) – Curso de Ciências da Computação, Universidade do Extremo Sul Catarinense, Criciúma, 2005.

CASSETTARI JÚNIOR, José Márcio. **O Método de Lógica Fuzzy pelo Algoritmo Gustafson-Kessel na Tarefa de Clusterização da *Shell* Orion *Data Mining Engine*.** 146 f. Trabalho de Conclusão de Curso (Bacharel em Ciências da Computação) – Curso de Ciências da Computação, Universidade do Extremo Sul Catarinense, Criciúma, 2008.

CUNICO, Luiz Homero Bastos. **Técnicas em *Data Mining* Aplicadas na Predição de Satisfação de Funcionários de uma Rede de Lojas do Comércio Varejista.** 110 f. Dissertação de mestrado, Pós-Graduação em Métodos Numéricos em Engenharia, Universidade Federal do Paraná, Curitiba, 2005. Disponível em http://www.dominiopublico.gov.br/pesquisa/DetalheObraForm.do?select_action=&co_obra=105573. Acesso em: 23 de setembro de 2009.

DONG, Guozhu; PEI, Jian. ***Sequence Data Mining*.** Springer, 2007. 150 p.

FAN, Hongjian. ***Efficient Mining of Interesting Emerging Patterns and Their Effective Use in Classification*.** 222 f. Tese de Doutorado (Doutor em Ciências da Computação) – Doutorado em Ciências da Computação, Universidade de Melbourne, Victoria, Austrália, 2004. Disponível em http://theses.library.uwa.edu.au/adt-WU2007.0216. Acesso em: 21 de setembro de 2008.

GARCIA-MOLINA, Hector; ULLMAN, Jeffrey D.; WIDOM, Jennifer. **Implementação de sistemas de bancos de dados.** Tradução Vandenberg D. de Souza. Rio de Janeiro: Campus, 2001. 685 p.

GOLDSCHMIDT, Ronaldo; PASSOS, Emmanuel. ***Data Mining:*** um guia prático. Rio de Janeiro: Elsevier, 2005.

HAN, Jiawei; KAMBER, Micheline. ***Data mining****: concepts and techniques.* San Francisco: Morgan Kaufmann Publishers, 2001. 550 p.

HELLERSTEIN, Joseph M.; STONEBRAKER, Michael (Eds.). ***Readings in Database Systems.*** Massachusetts: Massachusetts Institute of Technology, 2005.

HORSTMANN, Cay. ***Big Java*****.** Porto Alegre: Bookman, 2004. 1125 p.

INTELLIGENT SYSTEMS RESEARCH LLC. **ODBCMINE.** 2009.

JÜRJENS, Jan. ***Secure Systems: Development with UML.*** Alemãnha: Springer, 2005. 309 p.

KANTARDZIC, Mehmed. ***Data Mining:*** *Concepts, Models, Methods, and Algorithms*. John Wiley & Sons, 2003.

LAROSE, Daniel T. ***Discovering knowledge in data****: an introduction to data mining.* New Jersey: John Wiley & Sons, 2005.

LUGER, George F. **Inteligência artificial:** estruturas e estratégias para a solução de problemas complexos. Porto Alegre: Bookmann, 2004. 4ª ed.

MARTINS, Dênis Piazza. **O Algoritmo de Particionamento K-Means na Tarefa de Clusterização da *Shell* Orion *Data Mining Engine*.** 77 f. Trabalho de Conclusão de Curso (Bacharel em Ciências da Computação) – Curso de Ciências da Computação, Universidade do Extremo Sul Catarinense, Criciúma, 2007.

MITRA, Sushmita; ACHARYA, Tinku. ***Data mining****: multimedia, soft computing, and bioinformatics*. New Jersey: John Wiley & Sons, 2003.

MOTTA, Custódio G. L. da. **Sistema Inteligente para Avaliação de Riscos em Vias de Transporte Terrestre.** 2004. 153 f. - Programa de Pós-graduação de Engenharia, Universidade Federal do Rio de Janeiro. Disponível em: http://www.coc.ufrj.br/teses/mestrado/inter/2004/Teses/MOTTA_CGL_04_t_M_int.pdf Acesso em: 05 de agosto de 2009.

PELEGRIN, Diana Colombo. **A Tarefa de Classificação e o Algoritmo ID3 para Indução de Árvores de Decisão na *Shell* de *Data Mining* Orion.** 91 f. Trabalho de Conclusão de Curso (Bacharel em Ciências da Computação) – Curso de Ciências da Computação, Universidade do Extremo Sul Catarinense, Criciúma, 2005.

PEREGO, Daniel. **O Método de Lógica *Fuzzy* pelo Algoritmo Gath-Geva na Tarefa de Clusterização da *Shell* Orion *Data Mining Engine*.** 124 f. Trabalho de Conclusão

de Curso (Bacharel em Ciências da Computação) – Curso de Ciências da Computação, Universidade do Extremo Sul Catarinense, Criciúma, 2009.

PUJARI, Arun K. ***Data Mining Techniques.*** Índia: Universities Press, 2001.

QUINLAN, John Ross. ***C4.5:*** *Programs for Machine Learning.* Morgan Kaufmann Publishers, 1993.

RAIMUNDO, Lidiane Rosso. **O Algoritmo CART pelo Critério de GINI na Tarefa de Classificação da *Shell* Orion *Data Mining Engine*.** 106 f. Trabalho de Conclusão de Curso (Bacharel em Ciências da Computação) – Curso de Ciências da Computação, Universidade do Extremo Sul Catarinense, Criciúma, 2007.

REA, Sara Morgan. ***Building Intelligent .NET Applications***: *Agents, Data Mining, Rule-Based Systems, and Speech Processing*. *Addison Wesley Professional*, 2005.

REZENDE, Solange Oliveira (Coord.). **Sistemas Inteligentes:** Fundamentos e Aplicações. Barueri, SP: Manole, 2005. 525 p.

ROKACH, Lior; MAIMON, Oded. ***Data Mining with Decision Trees****: theory and applications*. *Singapore: World Scientific Publishing Co. Pte. Ltd*, 2008. 244 p.

RUD, Olivia Parr. ***Data Mining Cookbook***: *Modeling Data for Marketing, Risk, and Customer Relationship Management.* John Wiley & Sons, 2001.

SOUKUP, Tom; DAVIDSON, Ian. ***Visual Data Mining:*** *techniques and Tools for Data Visualization and Mining.* John Wiley & Sons, 2002. 389 p.

SOUTO, Marcílio de. ***Weka***: manual em português. Natal - RN, 2004. Disponível em: http://www.dimap.ufrn.br/~marcilio/IA/course-IA.htm. Acesso em: 15 de julho de 2009.

SYBASE MANUALS. Disponível em: http://manuals.sybase.com/onlinebooks. Acesso em 10 de março de 2009.

SYBASE SUCCESS STORIES. Disponível em: http://www.ianywhere.com/success_stories/sql_anywhere_success.html. Acesso em 15 de maio de 2009.

TANG, ZhaoHui, MACLENNAN, Jamie. ***Data Mining with SQL Server 2005.*** Indiana: Wiley Publishing, Inc., 2005.

TANIAR, David (Ed.). ***Data Mining and Knowledge Discovery Technologies*.** New York: IGI Publishing, 2008.

WAIKATO, University of, ***Weka 3***: *Data Mining Software in Java*. Waikato - New Zealand, 2005. Disponível em: http://www.cs.waikato.ac.nz/ml/weka. Acesso em: 29 de março de 2009.

WANG, Lipo; FU, Xiuju. ***Data Mining with Computational Intelligence.*** Alemãnha: Springer Verlag, 2005.

WITTEN, Ian H; FRANK, Eibe. ***Data mining****: practical machine learning tools and tecniques.* San Francisco: Morgan Kaufmann Publishers, 2005. 525 p. 2ª ed.

YE, Nong (Ed.). ***The handbook of data mining.*** New Jersey: Lawrence Erlbaum Associates, Inc., 2003.

ZANUSSO, Maria Bernadete. **Rede Difusa com T-Normas Diferenciáveis.** Departamento de Ciências Exatas e Tecnologia. Universidade Federal do Mato Grosso do Sul. 2002. 83 f. Disponível em: http://www.dct.ufms.br/~mzanusso/producao/CarMarAnd.pdf. Acesso em: 28 de outubro de 2009.